IntechOpen

Recent Trends in Computational Science and Engineering

Edited by M. Serdar Çelebi

RECENT TRENDS IN COMPUTATIONAL SCIENCE AND ENGINEERING

Edited by **M. Serdar Çelebi**

Recent Trends in Computational Science and Engineering
http://dx.doi.org/10.5772/intechopen.70387
Edited by M. Serdar Çelebi

Contributors

Sergey Voronin, Christophe Zaroli, Madina Mansurova, Darkhan Akhmed-Zaki, Timur Imankulov, Danil Lebedev, Olzhas Turar, Beimbet Daribayev, Sanzhar Aubakirov, Aday Shomanov, Victor Eduardo Martinez-Luaces, Mauricio Ohanian, Sakir Yucel, Frederick Ferguson, Julio Cesar Mendez, David Nii Darku Dodoo-Amoo, Frank Uhlig

Notice

Statements and opinions expressed in the chapters are these of the individual contributors and not necessarily those of the editors or publisher. No responsibility is accepted for the accuracy of information contained in the published chapters. The publisher assumes no responsibility for any damage or injury to persons or property arising out of the use of any materials, instructions, methods or ideas contained in the book.

First published in London, United Kingdom, 2018 by IntechOpen
eBook (PDF) Published by IntechOpen, 2019
IntechOpen is the global imprint of INTECHOPEN LIMITED, registered in England and Wales, registration number: 11086078, The Shard, 25th floor, 32 London Bridge Street
London, SE19SG – United Kingdom
Printed in Croatia

British Library Cataloguing-in-Publication Data
A catalogue record for this book is available from the British Library

Additional hard and PDF copies can be obtained from orders@intechopen.com

Recent Trends in Computational Science and Engineering
Edited by M. Serdar Çelebi

p. cm.

Print ISBN 978-1-78923-192-2

Online ISBN 978-1-78923-193-9

eBook (PDF) ISBN 978-1-83881-470-0

Meet the editor

Mustafa Serdar Çelebi, MSc, PD, PhD, works as a professor in the Department of Computational Science and Engineering, Istanbul Technical University (ITU), Turkey. He served in different academic levels ranging from program coordinator to deanery. He is the founding director of UHeM (National Center for High-Performance Computing of Turkey). Dr. Çelebi has been a referee in more than 15 different SCI/SCIE journals and an external expert in international projects and organizations such as FP7 ICT capacities, e-infrastructures, virtual communities, ORAU, and PAZY. He has published around 100 papers in journals, proceedings, and reports and presented about 45 talks at various meetings, workshops, seminars, and conferences. He organized 23 summer school workshops and conferences in the fields of high-performance computing, parallel programming, biomechanics, and computational fluid dynamics.

His fields of research are computational fluid dynamics, biomechanics, computational algebra, parallel numerical algorithms, medical image processing, and geometric modeling.

Contents

Preface

Computational science and engineering (CSE) is a broad multidisciplinary area including a variety of applications in science, engineering, numerical methods, applied mathematics, and computer science disciplines. Due to the recent technological developments in software and hardware, CSE's approach has a critical importance on the *integration* of knowledge and methodologies from all of these disciplines. Thus, it requires field expertise (science or engineering), mathematical modeling and its numerical analysis, algorithm design and its software implementation, model validation and post-processing, and interpretation of the data obtained by different visualization and animation tools.

It is widely accepted that simulation is the third pillar of science besides theory and experiment. Recently, computer-based models and computer simulations have become an important part of the research activities even in some cases supporting/replacing the experimentation. It is observed in many areas of science and engineering over the past 30 years that the boundary has been crossed where simulation or simulation in combination with experiment is more effective (in terms of time, cost, or accuracy) than experiment alone for actual science and engineering needs. One of the most powerful sides of CSE is that it enables the simulation of processes, nature, and complex systems that cannot be studied by real experiments because these experiments are too dangerous, too expensive, unethical, or just technically impossible.

With the breathtaking, long-standing, and continuing growth in speed, memory, and cost-effectiveness of computers, and with similar following improvements in software technology and numerical algorithms, the existing and future benefits of CSE are enormous and will be a crucial enabling technology in the future. Almost all major disciplines in science and engineering have their own computational interface now. CSE brings to all these disciplines a novel and innovative point of view with the help of growing emerging technologies in hardware, software, and numerical methods. Another important benefit of the CSE approach is to urge scientists and engineers to model with better realism using multi-scale, multi-phase, multi-physics, and multi-disciplinary approaches. Although multi-scale modeling has always been part of science, CSE has led to a renewed interest in it and has opened new perspectives and challenges. Concerning multi-physics and multi-disciplinary approaches, this motivates more and more the intelligent coupling of distinct, existing models: fluid dynamics plus structural mechanics models, thermal coupling or thermal plus electrical models, etc.

Considering all these benefits and potential interfaces and impacts to other disciplines in science and engineering, CSE has extremely wide and rich content and subtopics. With this book study, we tried to touch very few but prominent of these topics. The book provides a collection of different types of applications and visions to various disciplinary key aspects, which comprises both problem-driven and methodology-driven approaches at the same time.

This book is structured into the following six chapters:

Chapter 1 introduces the reader the importance of using computational and information technologies for both numerical models including complex initial boundary value problems and large unstructured and semi-structured data processing. In the first part of the chapter, the numerical model of the oil displacement problem was considered by injection of chemical reagents to increase oil recovery of reservoir. The second part of the chapter also describes parallel implementation of the document clustering algorithm used as a heuristic genetic algorithm. A novel UPC implementation of MapReduce framework for semi-structured data processing was introduced.

Chapter 2 outlines the epoch-making achievements and transformations that have occurred over time for matrix computations. More specifically, this valuable chapter focuses on how matrix concepts and algorithms have been developed from approximately 3000 BC to today and even tomorrow. This work separates this history to eight noticeable epochs that are distinguished from each other by the introduction of evolutionary new concepts and subsequent radically new computational methods, which are also extremely important to computational science and engineering discipline.

Chapter 3 presents prominent techniques in the field of inverse problems covering both classical and newer approaches. Presented methods are highly important and useful for the computational inverse problems. In the chapter, authors give the field practitioners an idea on when and how they can likely use these techniques. In particular, this chapter offers entries on the following materials:

- Matrix factorizations and sparse matrices
- Direct solutions and pivoted factorizations
- Least squares problems and regularization
- Non-linear least squares problems
- Low-rank matrix factorizations and randomized algorithms
- An introduction to Backus-Gilbert inversion

In **Chapter 4**, a hypersonic flat-plate problem is studied using a novel technique called integro-differential scheme (IDS) that combines the traditional finite volume and the finite difference methods under realistic conditions, at high Reynolds and Mach numbers. The numerical scheme implemented in this chapter solves the full unsteady Navier-Stokes Equations including mass, momentum, and energy conservations. The major remark of this chapter is to numerically solve the hypersonic flow over a flat plate with a novel numerical approach called IDS using suggested boundary conditions for the flow.

Chapter 5 addresses important key questions on a smart wireless network covering most of the smart city. This chapter introduces the smart community wireless platform as a conceptual approach rather than a technical study. Relevant dynamics in measuring the total cost, benefits, drawbacks, and risks of smart community wireless platforms are examined, and developed models for estimating the success of these platforms under various conditions and scenarios are surveyed. It outlines how the generic model could be instantiated for specific dynamics and to analyze different scenarios. The question of how the city could inspire and assist the communities to build their community wireless network and then coalesce them for a city-wide wireless network is also addressed.

Chapter 6 discusses the usage of electrochemical noise analysis to study potential fluctuations produced by kinetic variations along the corrosion process using the signal processing methods. Superimposing Gaussian noise on-trivial trend lines simulates particular signal data, as the first approach. Then, artificial intelligence for trend removal is used, as the second approach combining an interval signal processing with back propagation neural networks. This chapter particularly shows the increasing difference between trend and noise, and the artificial neural networks (ANN) became less accurate. In this chapter, moving median removal (MMR) yields the best results when polynomial fitting, moving average removal (MAR), and moving median removal (MMR) are compared.

I hope that this book provides the reader with a valuable source of idea for the solution of problems in different scientific and engineering disciplines and also serves as a motivation for the integrative or interdisciplinary study in computational science and engineering. Finally, I thank all the respectable contributors of this book for sharing their valuable experiences in different interesting and prominent fields of science and engineering.

Prof. Dr. M. Serdar Çelebi
Istanbul Technical University
Istanbul, Turkey

Chapter 1

High-Performance Computational and Information Technologies for Numerical Models and Data Processing

Darkhan Akhmed-Zaki, Madina Mansurova,
Timur Imankulov, Danil Lebedev, Olzhas Turar,
Beimbet Daribayev, Sanzhar Aubakirov,
Aday Shomanov and Kanat Aidarov

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.73836

Abstract

This chapter discusses high-performance computational and information technologies for numerical models and data processing. In the first part of the chapter, the numerical model of the oil displacement problem was considered by injection of chemical reagents to increase oil recovery of reservoir. Moreover the fragmented algorithm was developed for solving this problem and the algorithm for high-performance visualization of calculated data. Analysis and comparison of parallel algorithms based on the fragmented approach and using MPI technologies are also presented. The algorithm for solving given problem on mobile platforms and analysis of computational results is given too. In the second part of the chapter, the problem of unstructured and semi-structured data processing was considered. It was decided to address the task of n-gram extraction which requires a lot of computing with large amount of textual data. In order to deal with such complexity, there was a need to adopt and implement parallelization patterns. The second part of the chapter also describes parallel implementation of the document clustering algorithm that used a heuristic genetic algorithm. Finally, a novel UPC implementation of MapReduce framework for semi-structured data processing was introduced which allows to express data parallel applications using simple sequential code.

Keywords: fragmented algorithm, high-performance visualization, computational algorithms on mobile platforms, MPI, unstructured and semi-structured data processing, n-gram extraction, MapReduce framework

1. Introduction

With development of computer technology level and high-performance systems across the world, efficiency of solving problems in the field of fundamental and engineering research is increasing. Annual development of mathematical models allows to study physical and chemical processes in greater detail. Modern numerical methods are also being developed for solving applied problems and an amount of calculations are increasing. In this regard, using high-performance computing and computational technologies to solve applied problems with each year becomes more relevant.

In the middle stages of the development of high-viscosity oil fields, the problem of decreasing oil recovery becomes an issue. Increasing oil recovery of reservoirs remains one of urgent tasks at the moment. Methods of injecting polymers and surfactants into an oil reservoir are currently widely used in the oil industry and are considered as one of the effective methods for increasing the oil recovery of reservoirs [1, 2]. Therefore, the problem of oil displacement process by polymer and surfactant flooding was perceived as being a task for given working group.

Parallel implementation of the oil displacement problem and applied method appears to be complex problem of system parallel programming because it requires to provide synchronization of separate computational processes, network data transfer, etc. In order to decrease complexity of such parallel programs, technology of fragmented programming and its implementation called LuNA (Language for Numerical Algorithms) were adopted [3].

Visualization is an integral part of the analysis during the processing of the scientific data. It has a significant role in large-scale computational experiments on modern high-performance engines. The amount of data obtained in such calculations can reach several terabytes. Such system requires a well-designed and implemented client-side visualization module taking into account its client orientation. So such programming module was applied using modern visualization technology Vulkan API [4].

Nowadays full computational potential of mobile devices almost not used because of devices being idle for extended periods during a day. There are number of projects such as Berkeley Open Infrastructure for Network Computing (BOINC) which use excessive computational capabilities of PCs and mobile devices across the globe [5]. While provisioning services for its customers as integrator of numerous computational resources for solving their problems, the processing itself was conducted using only CPUs. Many recent mobile devices are equipped with powerful GPUs generally used for 3D graphics rendering. By efficient usage of mobile GPUs, one can achieve much more performance from a single device therefore increasing overall productivity of such integrational computations. This task requires the mobile software installed to be able to use capabilities offered by GPUs. Following researchers studied issues and possibilities related to exploit GPU capabilities of mobile devices in integrated computations: Zhao [6], Montella et al. [7].

Because of the rapid progress on computer-based communications and information dissemination, large amounts of data are daily generated and available in many domains. The purpose of the research presented in the second part of the chapter is to develop models and algorithms for unstructured and semi-structured data processing using high-performance parallel and distributed technologies.

Today huge amount of information are being associated with the web technology and the internet. To gather useful information from it, these text has to be categorized. Text categorization is a very important technique for text data mining and analytics. It is relevant to discovery various different kinds of knowledge. It is related to topic mining and analysis. It is also related to opinion mining and sentiment analysis, which has to do with knowledge discovery about the observer, the human sensor. The observer based on the content they produce can be categorized. The indexing influences the ease and effectiveness of a text categorization system [8]. The simplest indexing is formed by treating each word as a feature. However, words have properties, such as synonymy and polysemy. These have motivated attempts to use more complex feature extraction methods in text categorization tasks. If a syntactic parse of text is available, then features can be defined by the presence of two or more words in particular syntactic relationships. Nowadays authors [9–11] have used phrases (n-grams), rather than individual words, as indexing terms. In this work, the task was also addressed to n-gram text extraction which is a big computational problem when a large amount of textual data is given to process. In order to deal with such complexity, there was a need to adopt and implement parallelization patterns.

> *The chapter also focuses on research related to the application of genetic algorithm for document clustering. Genetic algorithms make it possible to take into account peculiarities of the search space by adjusting the parameters and selecting the best solutions from the solutions obtained by the population [12–14]. Clustering algorithm is based on the assessment of the similarity between objects in a competitive situation. Since clustering problem solution requires large computational resources parallelization on the stage of genetic algorithm for setting the coefficients in the formula of similarity measures was performed, as well as on the stage of data clustering.*

MapReduce technology has shown a great potential in dealing with large-scale data processing problems [15, 16]. Such batch-oriented MapReduce systems as Apache Hadoop, however, lacks efficiency in dealing with iterative problems. The main bottleneck can be attributed to slow disk operations arising in data storage after current iteration in a distributed file system. Number of solutions that deal with that problem has been proposed in a literature, including ones that propose novel techniques that optimize loops [17] and ones that try to keep static data [18]. Recently introduced novel approaches rely mostly on in-memory processing mechanisms [19, 20]. Also some types of data parallel problems require efficient communication between parallel workers in order to be able to implement specific nature of the data exchange patterns. In such a way, it is necessary to consider other parallel programming models that can be effectively combined with MapReduce.

Partitioned global address space (PGAS) model presents an interesting approach to deal with data communication problem. In PGAS model, a global memory is divided among threads with different choices of memory to thread mappings. Several works introduced different approaches to implement MapReduce functionality in a frame of PGAS model. For example, in [21], authors introduce a design of MapReduce system based on using unified parallel C that belongs to a family of PGAS languages. In that approach collective operations for data exchange are employed. A different implementation of MapReduce based on X-10 parallel programming language of PGAS family uses hashmap data structure to deal with data exchange task [22].

2. Mathematical and computer modeling of 3D oil displacement process in porous media

2.1. Mathematical model of polymer and surfactant flooding

In general processes of oil displacement by chemical reagents controlled by complex physical and chemical processes. Therefore, exact simulation of such processes using numerical methods produces a number of certain issues. Therefore, the mathematical model of two-phase flow in porous media has the following assumptions: (1) flow is incompressible; (2) gravitational forces and capillary effects are neglected and (3) two-phase flow (water and oil) obeys Darcy's law.

Taking into account the foregoing assumptions, a system of equation was written for two-phase flow in porous media, which contains mass conservation equations for water and oil phases, the Darcy's law, and the equation for the transfer of concentration and salt in the reservoir [23, 24].

Mass conservation equations can be written as follows:

$$m\frac{\partial S_w}{\partial t} + div(\boldsymbol{v_w}) = q_1 \tag{1}$$

$$m\frac{\partial S_o}{\partial t} + div(\boldsymbol{v_o}) = q_2 \tag{2}$$

where m is the porosity, S_w, S_o are the water and oil saturations, q_1, q_2 are the source or sink. Porous medium saturated with fluids: $S_w + S_o = 1$.

Velocities of each phases is given by Darcy's law:

$$\boldsymbol{v_i} = -K_0\frac{f_i(s)}{\mu_i}\nabla P, \quad i = w, o \tag{3}$$

where $f_i(s)$, μ_i is the relative permeability and viscosity for phase i, P is the pressure, K_0 is the permeability tensor.

Polymer, surfactant, salt and heat transport equations are given by:

$$m\frac{\partial}{\partial t}\left(c_p s_w\right) + \frac{\partial a_p}{\partial t} + div\left(\boldsymbol{v_w} c_p\right) = div\left(mD_{pw} s_w \nabla c_p\right) \tag{4}$$

$$m\frac{\partial}{\partial t}(c_{sw}s_w + c_{so}s_o) + \frac{\partial a_{surf}}{\partial t} + div(\boldsymbol{v_w}c_{sw} + \boldsymbol{v_o}c_{so}) = div(mD_{sw}s_w\nabla c_{sw} + mD_{so}s_o\nabla c_{so}) \qquad (5)$$

$$m\frac{\partial}{\partial t}(c_s s_w) + div(\boldsymbol{v_w}c_s) = 0 \qquad (6)$$

$$\frac{\partial}{\partial t}[(1-m)C_r\rho_r T + m(C_w s_w \rho_w + C_0 s_0 \rho_0)T] + div(\rho_w C_w \boldsymbol{v}_w T + \rho_0 C_0 \boldsymbol{v}_0 T) = div(D\nabla T),$$

$$D = (1-m)\lambda_0 + m(\lambda_1 s_w + \lambda_2 s_0) \qquad (7)$$

where c_p, c_s is the concentrations of polymer and salt in water phase, c_{sw}, c_{so} are the concentration of surfactant in water and oleic phases, a_p, a_{surf} are the adsorption functions, D_{pw}, D_{sw}, D_{so} are the diffusion coefficients, C_w, C_o, C_r are the specific heat of water, oil and rock, ρ_w, ρ_o, ρ_r are the water, oil and rock densities, $\lambda_0, \lambda_1, \lambda_2$ are the thermal conductivity coefficients.

Initial conditions:

$$s_w|_{t=0} = s_{w0}, \, c_{pw}\big|_{t=0} = c_{p0}, \, c_s|_{t=0} = c_{s0}, \, c_{sw}|_{t=0} = c_{sw0}, \quad c_{so}|_{t=0} = c_{so0},$$

$$T|_{t=0} = T_p, \, a_{surf0}\big|_{t=0} = a_{surf0}, \quad a_p\big|_{t=0} = a_{p0} \qquad (8)$$

Boundary conditions:

$$\left.\frac{\partial s_w}{\partial n}\right|_{\partial\Omega} = 0; \qquad \left.\frac{\partial P}{\partial n}\right|_{\partial\Omega} = 0; \qquad \left.\frac{\partial T}{\partial n}\right|_{\partial\Omega} = 0; \left.\frac{\partial c_{pw}}{\partial n}\right|_{\partial\Omega} = 0;$$

$$\left.\frac{\partial c_{pw}}{\partial n}\right|_{\partial\Omega} = 0; \quad \left.\frac{\partial c_{sw}}{\partial n}\right|_{\partial\Omega} = 0; \quad \left.\frac{\partial c_s}{\partial n}\right|_{\partial\Omega} = 0 \qquad (9)$$

The following viscosity dependence on injected reagent concentrations and temperature was used:

$$\mu_a = \mu_w\left[1 + \left(\gamma_1 c_p + \gamma_2 c_p^2 + \gamma_3 c_{sw} + \gamma_4 c_{sw}^2\right)c_s^{\gamma_5} - \gamma_6(T - T_p)\right] \qquad (10)$$

$$\mu_o = \mu o_o[1 - \gamma_7(T - T_p)] \qquad (11)$$

where $\gamma_1, \gamma_2, \gamma_3, \gamma_4, \gamma_5, \gamma_6, \gamma_7$ are the constants, μo_o is the initial viscosity of oil phase, T_p is the reservoir temperature. The imbibition relative permeability curve for water/oil flow is given by

$$f_w(S_w) = S_w^{3.5}; \; f_o(S_w) = (1 - S_w)^{3.5} \qquad (12)$$

The process of displacement of oil by polymer and surfactant solutions described through developed mathematical model. First oil reservoir filled with surfactant solution is driven by conventional water. After that polymer solution injected in order to control the slug which improves volumetric sweep efficiency. This procedure followed by injection of an ordinary water flow. The amount of surfactant, polymer and water injected must be computed through developing mathematical model describing distributing of pressure and temperature, saturation of each phase, chemical concentrations of the process flowing within a reservoir. Reservoir dimensions and shape described within mathematical model as three-dimensional computational domain (**Figure 1a**).

The numerical solution of Eqs. (1)–(12) based on finite difference method and explicit/implicit scheme. The algorithm for constructing a solution is reduced to the following. The temperature of the reservoir and the injected water, the initial oil saturation of the reservoir, the initial pressure distribution, the technological and physical parameters of the reservoir are set. The values of saturation, pressure, concentration and temperature are solved according to the explicit Jacobi scheme on the basis of the finite difference method in the three-dimensional grid [25] (**Figure 1b**).

2.2. Fragmented algorithm

For solving the three-dimensional fluid flow problem, the method with stabilizing correction was used [26]. For implementation of parallel algorithm initial area (Nx × Ny × Nz) divided to subdomains. At first division, division made by z-axis where number of subdomains depends on number of processes, size of subarea equals Nz/size+2 shadow edges.

After that computations for the first and the second intermediary step were made in order to find pressures and saturations by algorithm described in previous section. Then the second subdivision of initial domain was done by y-axis and compute values of gas pressure by third step of the method. After the third step, boundaries for all variables were exchanged and compute first step of the method for further time step. At the end of this step domain was made subdivision again but already by x-axis and compute the second and the third steps. After that subdivision was made on by z-axis again, exchange shadow edges and start to compute first and second intermediary steps as shown in **Figure 2**.

Advantage of such scheme of initial three-dimensional domain division at computing two intermediary steps is the possibility to solve independently at each process by sequential sweep [27] for which there is no efficient parallel algorithm. But there are global communications after each second intermediary step which appears when initial domain is divided.

Testing conducted on “MVC” Supercomputer of Unified Supercomputer center of the Russian Academy of Sciences [28] which include nodes with two Xeon E5-2690 processors,

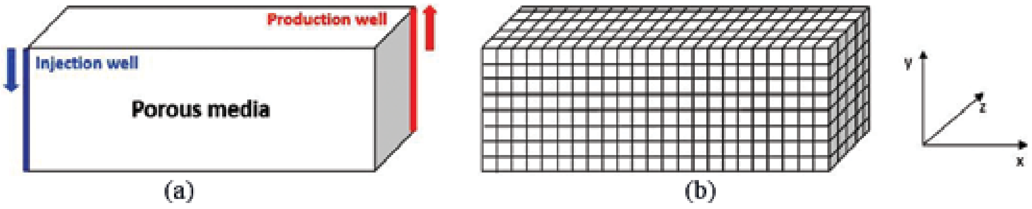


Figure 1. Computational domain (a) and computational grid (b).

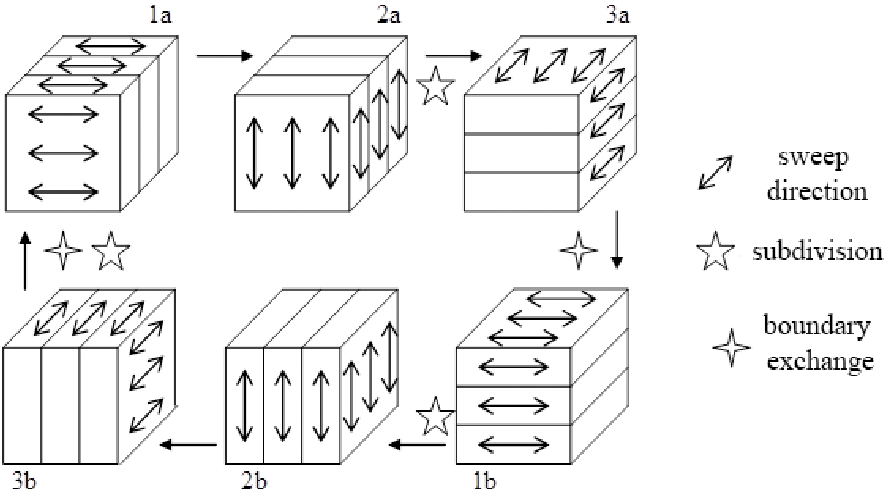


Figure 2. Scheme of computations for three-dimensional fluid flow problem.

communicational and transport network based on FDR Infiniband and 64 Gb of Operating Memory for each node. MPI MPICH3 and GCC compiler used.

Series of experiments allowed to define weak scalability of the implementation. That is for different problem sizes increase in problem size was proportional to number of computational nodes. Ideally, computation time must be equal for every experiment because number of computations in each node is approximately the same. In reality, increased time leads to increase in communication length and size of transferred values.

As show in **Figure 3** (x-axis shows number of processes and appropriate domain size), MPI implementation possesses best efficiency because unlike fragmented program it does not have overhead expenses belonging to LuNA system algorithms [29]. But efficiency does not reach 100% because of existing global communications appearing at decomposition of the domain in a process. Moreover, it can be noted that LuNA implementation approximately 200 time slower than MPI while LuNA with manual setting (LuNA-Fw) approximately 10 times slower.

From **Figure 3**, it can be seen that with increase of the problem size execution time for the sequential program disappears. This related to the fact that program data no longer fits to a memory of a single node while parallel and fragmented programs still do.

2.3. High-performance visualization

Let us consider the visualization module. Highly optimized visualization API with cross-platform support is needed. Previously only the OpenGL standard can be such tool. However, OpenGL has a number of limitations, mainly related to its high-level implementation. Because of this, it cannot use advantages of processors from different manufacturers.

At the moment, a new low-level visualization standard, the ideological continuation of OpenGL, Vulkan API [4] is rapidly developing. It also contains a functional for parallelizing

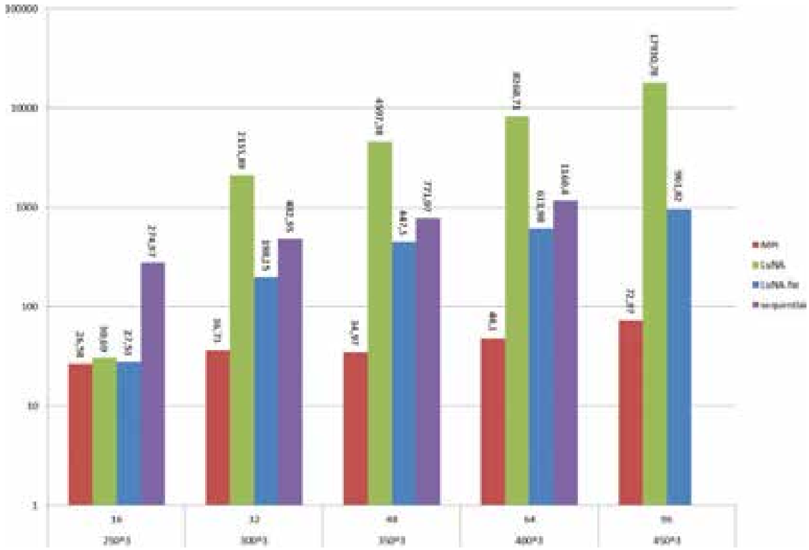


Figure 3. Weakly scalable testing. Dependence of computation time from the size of a problem and processes.

CPU-side computations and provides multiple performance improvements by reducing the number of CPU addressing using a technology similar to AMD Mantle [30]. Vulkan is a cross-platform response from the Khronos Group to the latest DirectX 12 Direct3D standard [31] developed by Microsoft and released with the Microsoft Windows 10 operating system.

The Vulkan toolkit is extremely low-level and most settings are manually configured. The main reason of the high performance shown by Vulkan compared to OpenGL is due to the decrease of the dependence of video processor on the CPU. Indeed, in the old visualization tools, the drawing of each animation frame was each time run directly from the CPU. Thus, after each rendering iteration and presentation to the screen, a signal was sent to the CPU, after which the video processor waited the completion of current CPU commands before launching the next iteration. In other words, CPU and GPU were synchronized on each call of the render function.

In an application that uses the Vulkan API, parts that flow on the CPU and the video processor are generally unsynchronized. Synchronization at the moments of necessity is controlled by the application itself, not by the driver and the library, as it was in OpenGL.

The demonstration of the implemented module is shown in **Figure 4**.

To test the capabilities of this visualizer, special test models obtained from the models presented above were used (**Figure 5**). To simplify the creation of model, colors of the cells were generated randomly. From this basic model (**Figure 5a**), using a special generator, a similar model was created consisting of a much larger number of polygons and active cells. Test models were generated by splitting each polygon of the base model.

The model shown in **Figure 5b** has a surface consisting of 62,078,400 polygons. The geometry of this model occupies 1420 MB or 1.387 GB on graphics memory. One can interact with this

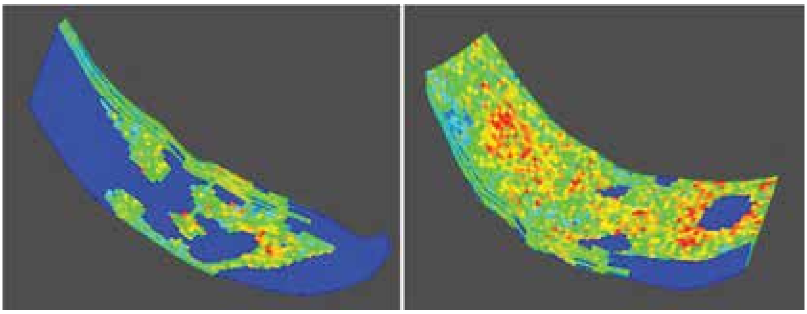


Figure 4. Demonstration of the visualization module using the Vulkan API (different colors represent the values of permeability along the Ox axis: red color for maximum values and blue for minimum).

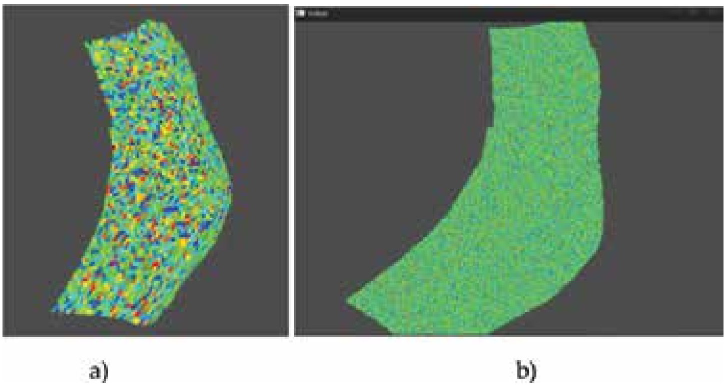


Figure 5. Basic model containing 67,165 active cells (a) coarse mesh; (b) finer mesh.

model in real time, since the rendering is done at a frequency of 51–53 frames per second. At the moment, Vulkan standard allows to significantly optimize the performance of graphics applications due to special technologies for working with data and video card resources at a low level.

By using the described technology, a new version of the information system visualizer shows a significant increase in drawing performance. The presented results of the rendering speed can theoretically correspond to the models with hundreds of millions of computational cells.

2.4. Computational algorithms on mobile platforms

2.4.1. Creation high-performance software on mobile devices

Nowadays, rapid grows of number of mobile devices pushed mobile industry to the very top of the global technological market making it one of the most important areas of public services. Huge interest in mobile market from common population set technological trend of mobile industry to a fastest possible route. CPUs and especially GPUs present in modern mobile devices being absolutely separate computational units can be used as a parts of heterogeneous computational platform combining ordinary servers and other alternative connected devices with purpose of integral computation. Recent models of mobile devices equipped with GPUs supporting nVidia CUDA technology. This technology can be used to implement parallel versions of conventional algorithms further allowing to solve computational intensive tasks. The mobile nature of computational devices allows to exploit them directly at oil field even if there no wireless connection. In case if there is access to digital network they can be a part of heterogeneous computational cluster.

This section describes oil displacement problem in order to test the parallel algorithm on GPU, which uses a shared memory for storage of a grid node values and comprises of various comparative tests focusing on effectiveness of mentioned algorithms. Oil displacement problem by polymer and surfactant injection taking into account temperature effects. The computer model described the complex real industrial problem of oil recovery [32, 33].

2.4.2. A parallel algorithm using CUDA technology on a mobile platform

Let us make assumption that GPU grids chosen for allocating program data contain several blocks. Every block represented in a three-dimensional form and program data copied from the global memory to the shared memory of a GPU during computation. After relocation of data into shared memory it cannot be used again. Therefore, it will be copied back from the global memory which usually appears to be slowest one. It means that copying data from the global memory to GPUs shared memory four times creates inefficiency. Other issue is that each subdomain requires data from its neighbors to continue computation. This creates situation where data will be copied from the global memory every time the boundary layer data needed [34].

To tackle abovementioned issue, the kernel function introduced into parallel implementation of an algorithm. Using kernel function allows data to be declared in a shared memory according to a size of a problem. Every thread within a single block has access to the shared memory only. The performance of the shared memory much higher than of the global memory. This allows to avoid loading data from the global memory every time the boundary data required which leads to noticeable increase in performance. The parallel algorithm within the kernel function works as follows: a temporary array declared in the shared memory where an output of a calculation will be stored; at first this array represent a copy of an input array; after that values at edges of a given array will be replaced by boundary values from neighboring blocks. Every time algorithm requires input data values it gets them from the shared memory instead of the global memory. High performance of the shared memory significantly speeds up total computation. One must be careful when working with boundary values from neighboring block because wrong choice of indices lead to incorrect output data.

Conducting mobile computations on problems related to real-world technological processes recently became popular topic among science and industry. Game industry actively utilize computaional capabilities of recent mobile devices by developing games with high-performance graphical data processing. Other examples are image recognition and machine learning in mobile cloud services. One can easily notice rapid grows of computational capabilities of modern mobile devices. Recent developments in this industry like nVidia Tegra X1 chipset possesses 256

GPU cores based on nVidia Maxwell architecture [35]. This chipset has extraordinary computational performance potential up to 1 TFlops which is comparable with performance of small supercomputer. Such situation undoubtedly expands area of problems solving with such devices to a new height.

2.4.3. Results and comparison of computational experiments

In **Figure 6**, the ratio of calculating time for solving oil recovery problem on the PC and the mobile device can be seen. By increasing the grid size, the time ratio decreases.

Figure 7 demonstrates the application for the mobile device on the base of the model of oil displacement process by polymer and surfactant taking into account the salinity and temperature of the reservoir in porous medium.

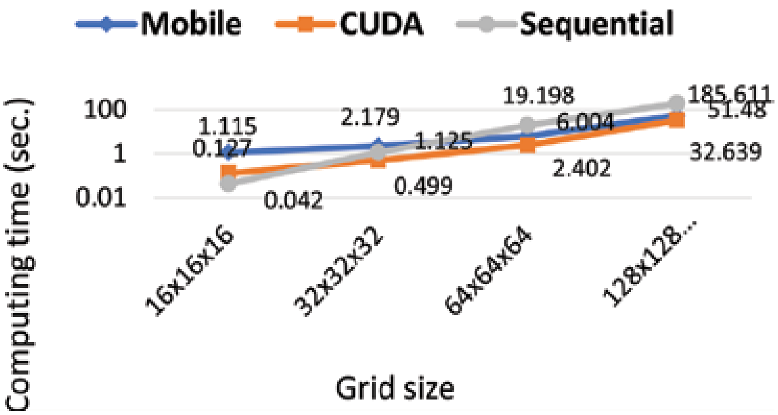


Figure 6. Computing times of mobile device and PC (polymer and surfactant flooding).

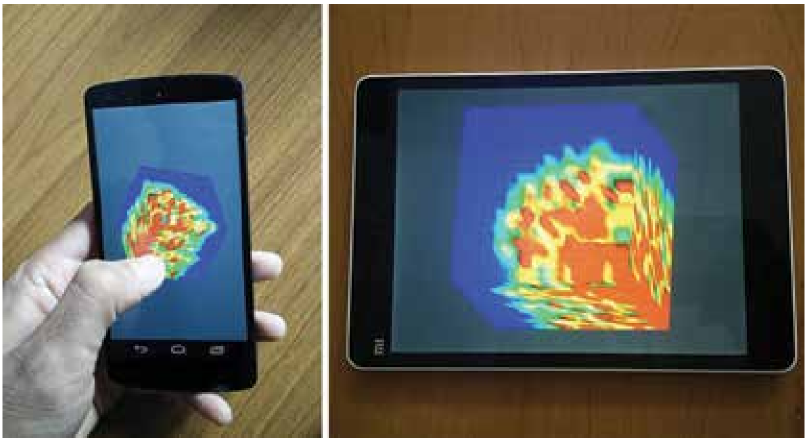


Figure 7. Demonstration of the mobile application results.

The prototype of hydrodynamic simulator developed for high-performance computations on mobile devices and uses existing industrial file formats of a known foreign software companies (Schlumberger, Roxar, etc.). This means that computation results of developed simulator can be backward compatible with file formats used in other software products by these companies. Advantage of a mobile application before traditional one is that the mobile app allows users get results of a computation being located directly on the field.

3. High-performance information technologies for data processing

3.1. Comparison of distributed computing approaches to complexity of n-gram extraction

Nowadays there are several HPC frameworks and platforms that can be used for the distributed computing text processing. n-Gram extracting task was implemented on three platforms: (1) MPJ Express, (2) Apache Hadoop, and (3) Apache Spark. Moreover two different kinds of the input datasets were used: (i) small number of large textual files and (ii) large number of small textual files. Experiments were conducted with each of the HPC platform, each experiment uses both datasets and the experiment repeats for a set of different file sizes. The speedup and efficiency among MPJ Express, Apache Hadoop, and Apache Spark were computed. The guidelines for choosing the platform could be provided based on the following criteria: kind of dataset (i) or (ii), dataset size, granularity of the input data, priority to reliability, or speedup. The contributions from our work include:

- Comprehensive experimental evaluation on English Wikipedia articles corpora;
- Time and space comparison between implementations on MPJ Express, Apache Hadoop, and Apache Spark;
- Detailed guidelines for choosing platform.

The n-gram feature extraction was conducted from the Wikipedia articles corpora. The corpora size is 4 gb and it is consists of more than 200000 articles, each article's size is approximately 20 kb. Furthermore all dataset was divided into six subsets: 64, 256, 512, 1024, 2048 and 4096 Mb, where each subset is divided into two sets: (i) a large number of small textual files and (ii) a small number of large textual files. The articles were kept as is for data set (i), whereas articles were concatenated into bigger files for data set (ii).

Our goal is to extract n-gram from all articles and from each article separately. So the full n-gram model was considered to be all extracted n-grams, where $n \in [1, k]$ and k are the length of longest sentence in the dataset. Further the method that is described by Google in their paper [36] was used and improvements suggested by work [37] were considered. Both algorithms are based on MapReduce paradigm. Method proposed by [37] optimized memory consumption overall performance, but at the same time rejecting not frequent n-grams. The reason of using Google's proposed algorithm is because our goal is to obtain full n-gram model.

As a result the algorithm for our goal of n-gram extraction was adopted from the individual articles. Our method operates with sentences, text of the articles is represented as set of sentences S, where $S = (S_1, S_2, \ldots, S_n)$ and each sentence S_n is a list of words $S_n = (W_1, W_2, \ldots, W_m)$, where W_m is a single word.

The functions sliding(), *map*(), and *reduce*() were implemented. Function *map*() takes list of sentences S and for each S_i executes sliding() function with the parameter $n = (0, 1, 2, \ldots, m)$, where n is size of slides (n-grams) that function will produce and m is number of words W in sentence S_i. Function *reduce*() takes output of *map*() function, which is the list of n-grams (list of list of words) and count similar ones. As a results it returns list of objects (n-gram; v), that is usually called Map, where v is the frequency of particular n-gram in the text. This approach provide ability to execute independent *map*() and avoid communication between nodes until *reduce*() stage.

For our experiments the cluster of 16 nodes was used, each node has the same characteristics. More details about cluster and frameworks could be found in [38] work. **Figure 8** shows results of the conducted experiments. It is clear that parallelization reveal good efficiency and speedup on all three HPC platforms. During our experiments, Apache Hadoop shows low speed and efficiency for a large number of small files. Researches [39] show that Apache Hadoop works faster if input data is represented as a few big files instead of many small files. This is because of HDFS design, which was developed for processing big data streams. Readings of many small files leads to many communications between nodes, many disk head movements and as a consequence leads to extremely inefficient work of HDFS. Details of the comparison could be found in the work [38].

3.2. Parallel text document clustering based on genetic algorithm

This section describes parallel implementation of the text document clustering algorithm. The algorithm is based on evaluation of the similarity between objects in a competitive situation, which leads to the notion of the function of rival similarity. While attributes of bibliographic description of scientific articles were chosen as the scales for determining similarity measure. A genetic algorithm is developed to find the weighting coefficients which are used in the formula of similarity measure. To speed up the performance of the algorithm, parallel computing technologies are used. Parallelization is executed in two stages: in the stage of the genetic algorithm, as well as directly in clustering. The parallel genetic algorithm is implemented with the help of MPJ Express library and the parallel clustering algorithm using the Java 8 Streams library. The results of computational experiments showing benefits of the parallel implementation of the algorithm are presented.

3.2.1. Clustering using the function of rival similarity

FRiS-Tax algorithm described in [40] is chosen as a clustering algorithm. The measure of rival similarity is introduced as follows. In the case of the given absolute value of similarity m(x, y) between two objects, the rival similarity of object a with object b on competition with c is calculated by the following formula:

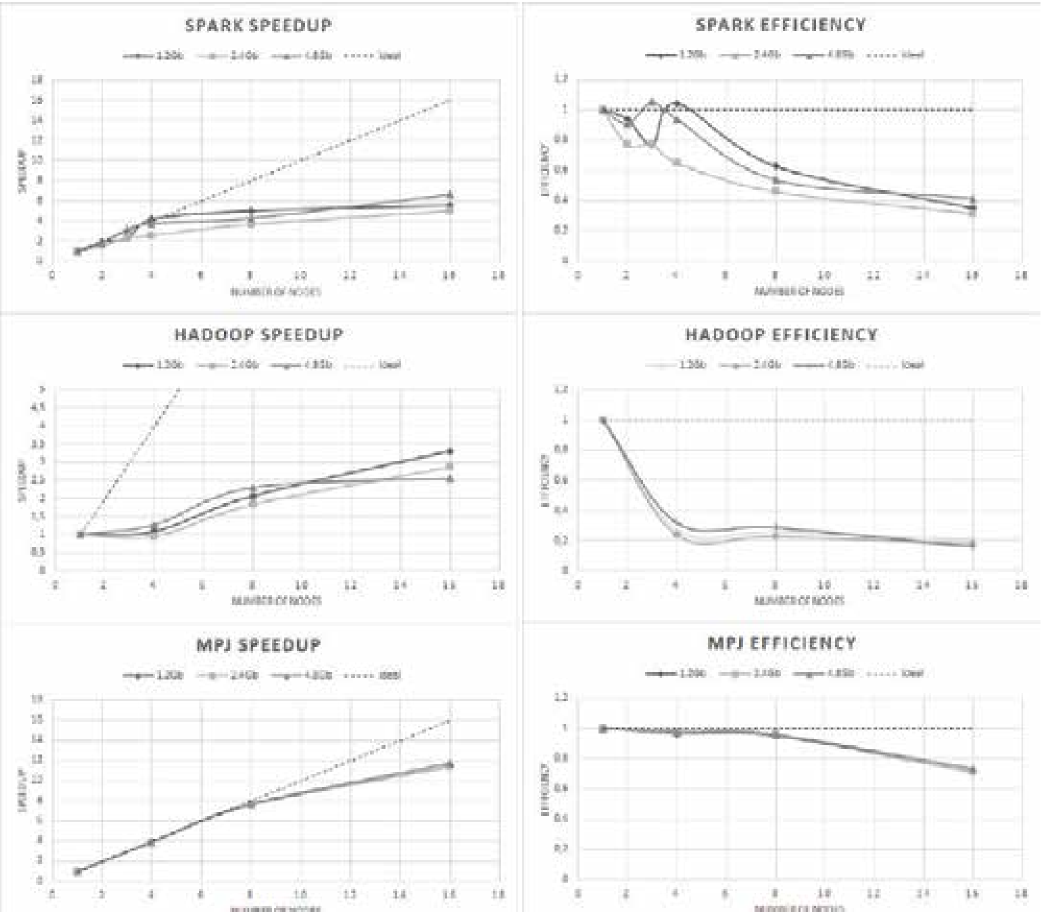


Figure 8. Speedup and efficiency of each platform.

$$F_{b/c}(a) = \frac{\mathrm{m(a,b)} - \mathrm{m(a,c)}}{\mathrm{m(a,b)} + \mathrm{m(a,c)}} \tag{13}$$

where F is called a function of rival similarity or FRiS-function. To measure similarity, the attributes of the bibliographic descriptions of scientific articles were proposed to be taken as scales.

The year of issue; code UDC; key words; authors; series; annotation; title were chosen as attributes of division of articles from bibliographic databases into clusters. A genetic algorithm was developed to choose weighting coefficients which are used in the formula of similarity measure (Eq. (13)). The use of genetic algorithm allows automating the search for the most acceptable weighting coefficients in the formula of similarity measure.

3.2.2. Genetic algorithm for adjustment of coefficients in the formula of similarity measure

To create the initial population of genetic algorithm and its further evolution, it is necessary to have an ordered chain of genes or a genotype. For this task, a chain of genes has a fixed length equal to 13 and presents a set of parameters made up on the basis of attributes of bibliographic description of documents.

In genetic algorithms, the individuals entering the population are presented by ordered subsequent genes or chromosomes with coded in them sets of the problem parameters.

At the stage of selection, the parents of the future individual are determined with the help of methods Roulette Selection, Tournament Selection, and Elitism Selection. The survived individuals take part in reproduction. For crossover operator, the following methods are used: One point crossover, Two point crossover, Uniform crossover, and Variable to Variable crossover. The stage of mutation is necessary not to let the solution of the problem get into a local extremum. It is supposed that, after the crossover is completed, part of the new individuals undergo mutations. In our case, 25% of all individuals are selected which are subjected to mutation. In this work, the quality of the obtained clusters is evaluated using the Purity and Root mean square deviation measures of estimation.

3.2.3. *Development of the parallel clustering algorithm*

Parallelization is carried out in two stages of the algorithm of clustering. The first stage is occurred during the selection of individuals in the genetic algorithm when clustering is performed with different sets of weighting coefficients. The program is written in Java, and this stage of the parallel algorithm is performed using MPJ Express. Secondly, it is directly in the course of performing the clustering algorithm.

The load test revealed the two slowest stages in the clustering algorithm. They are the methods of finding the first centroid called pillar and finding the next pillar, which are doing $N^*(N\text{-}1)$ and $N^*(N\text{-}1)^*M$ operations, where N is the number of articles and M is the number of already found pillars. To accelerate these methods, the technology *Java* 8 *Streams* was used. Since repeated $(N\text{-}1)$ and $(N\text{-}1)^*M$ times operations in methods finding first and finding next pillar, respectively, are simple and their result need to be summarized at the end, it is reasonable to implement here *parallel*() method of *Java* 8 (**Figure 9**).

For clustering and performance analysis, the journal "Bulletin of KazNU" of 2008–2015 was used as initial data. Sampling includes 95 pdf documents. The total number of articles is 2837. The choice of the initial data is conditioned by the fact that all documents were divided into series (mathematics, biology, philosophy, etc.) and further divisions do not cause difficulties, when using measures of similarity based on only bibliographic descriptions or titles of the articles. In order to evaluate the quality of division of sampling, this body was divided into clusters with the help of an expert into the problem domain.

The time of execution was determined as follows. The measurements of the time of clustering processes were made for the clusters being formed on one computer node and several computer nodes for parallel realization. **Figures 10** and **11** present acceleration and efficiency of parallel realization. As it is seen in the constructed diagrams, with the increase in the number of processes, acceleration increases to a certain value which is related to the expenditure of communication. The most optimum number of processes proved to be eight at which the maximum value of acceleration was observed but the highest value of efficiency was achieved with 4 processes.

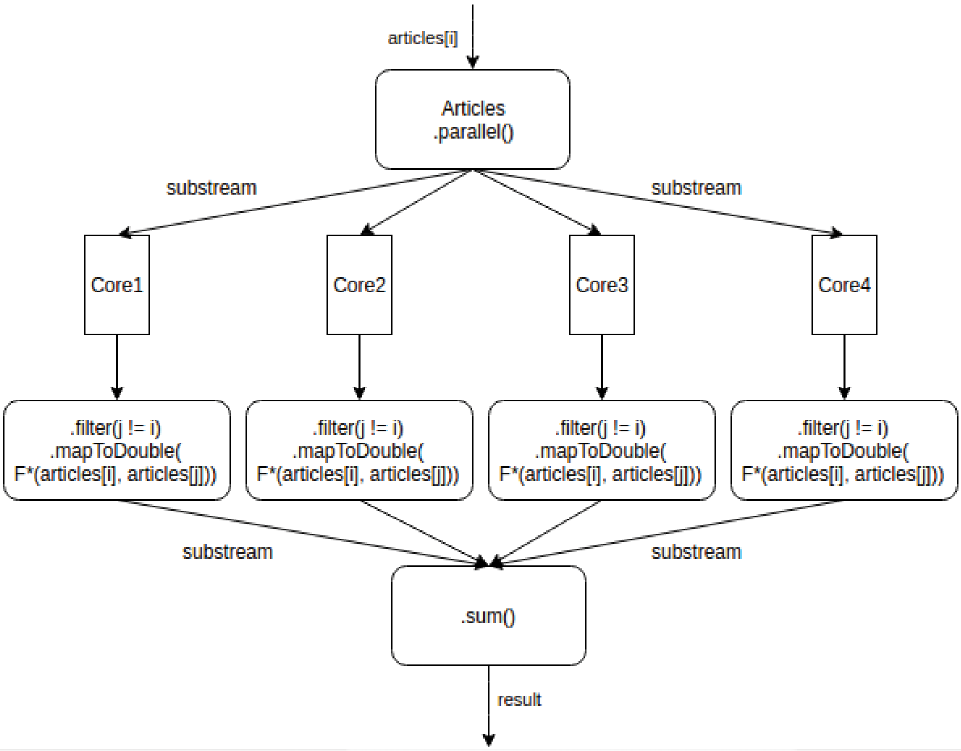


Figure 9. Stream parallelization on 4-core processor, find first pillar.

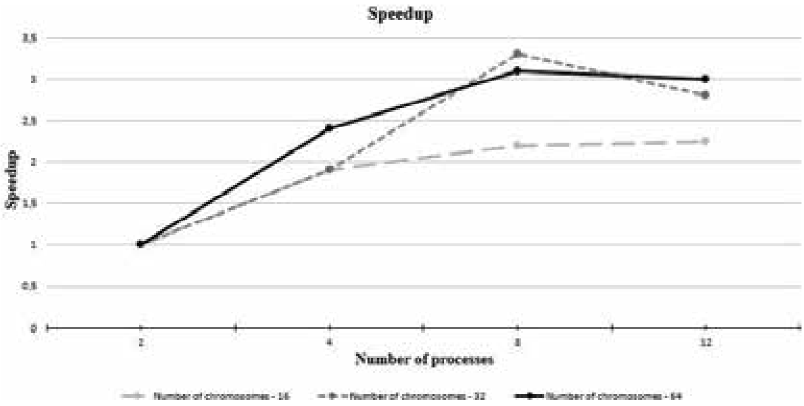


Figure 10. Speedup of parallel clustering algorithm.

It can be concluded that the use of the genetic algorithm allowed to determine the values of attributes at which clustering of documents gives the best results [41].

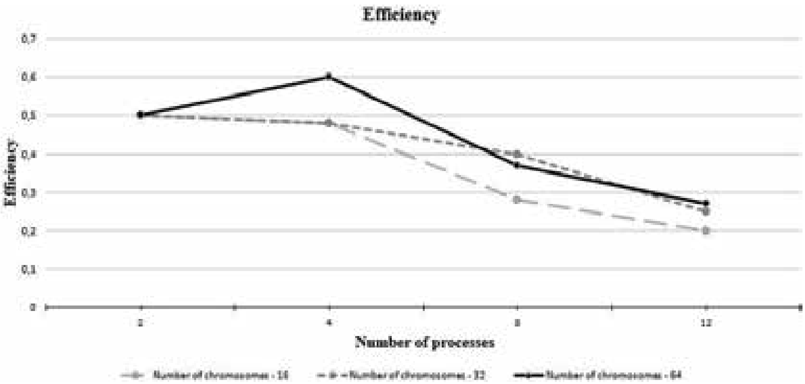


Figure 11. Efficiency of parallel clustering algorithm.

3.3. PGAS approach to implement MapReduce framework based on UPC language

In Section 3.1, the important role of the MapReduce paradigm and distributed file systems in large data processing tasks was emphasized. The weak side of distributed file systems is the considerable time spent on performing read and write operations. In this chapter, an approach to implement MapReduce functionality was described using partitioned global address space model (PGAS). PGAS is a parallel programming model in which memory is divided among threads with certain affinity rules. The affinity is a property that tells how memory is distributed among threads. In some sense, PGAS is considered to be a model that shares the properties of both shared and distributed memory models. The memory is divided in such a way that each thread controls some portion of shared memory region and a private memory which is used to store local to that thread variables. The obvious benefits of using such a model are:

- Transparent view of shared memory by each thread;
- No need to use low-level message passing techniques to exchange data between threads.

The implementation of MapReduce using PGAS approach is based on using hashmap data structure. Hashmap data structure is used to store key/value pairs generated during MapReduce execution. The main idea is that array containing hashmap entries is created in a global shared memory space. Array is distributed in such a way that each array entry correspond to exactly one thread. Since, hashmap is located in a global shared memory region each thread can view and modify/read the hashmap entries of the other threads. This way data exchange for the MapReduce (see **Figure 12**) can easily be implemented by just exchanging and distributing corresponding key/value pairs among different threads. For each thread to decide set of keys to be processed in reduce stage the problem of key distribution was formulated.

The problem of key distribution among threads after map stage has been stated in the following way:

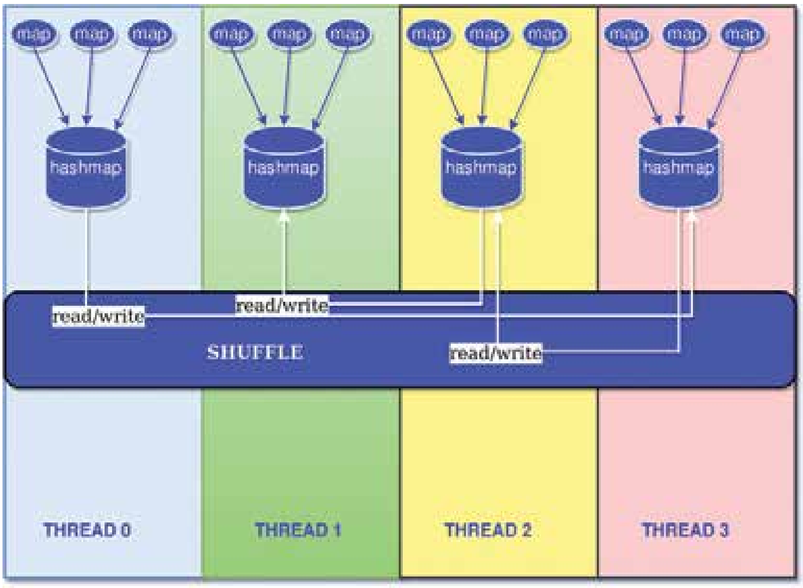


Figure 12. Data exchange mechanism for MapReduce using PGAS approach.

$$min \sum_{i=0}^{threads-1} \sum_{j=1}^{keys} x_{ij} \times cost_{ij} \tag{14}$$

$$x_{ij} \in \{0,1\} \tag{15}$$

$$min\left(\max_{i,j=0..threads-1} \left| load_i - load_j \right| \right) \tag{16}$$

$$load_i = \sum_{t=0}^{threads-1} \sum_{j=1}^{keys} x_{ij} \times size_{tj} \tag{17}$$

Finding the cost of assigning key j to thread i is done by building the *cost* matrix. The quantity $cost_{ij}$ represents the cost of moving key j to thread i. This value is defined to be a number of elements with certain key to be moved from other threads to the thread with an index of i. Keys need to be distributed in such a way that Eqs. (14) and (16) are satisfied. Eq. (15) specifies the domain of x_{ij}. The value of x_{ij} is equal to zero when thread i is not assigned to process key j and $x_{ij} = 1$ otherwise. Load balancing function is defined in Eq. (16) and can be computed as a minimum of maximal difference of loads assigned to any pair of threads. Load for each thread i is defined in Eq. (17) [42].

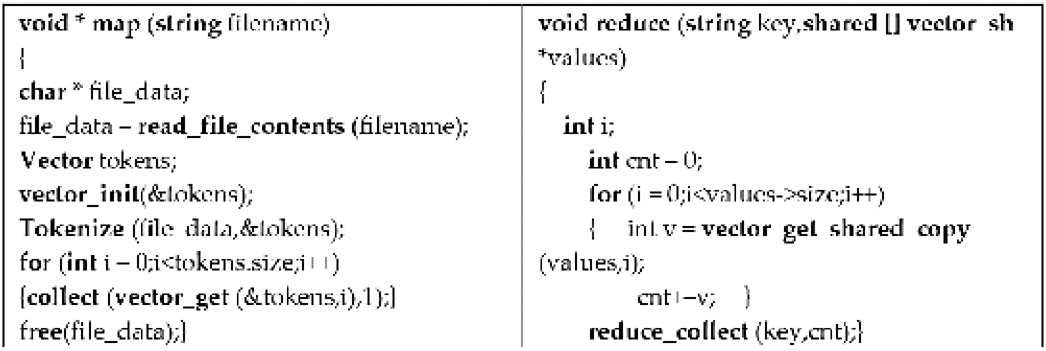
Since the described problem of key distribution is proven to be NP-hard, finding the optimal distribution even for a small set of keys is a computationally very expensive task. Therefore, a heuristic genetic algorithm was used that tries to find a close approximation to the optimal result.

The MapReduce framework has been tested on WordCount application (see **Figure 13**). WordCount application is used to compute number of occurrences of each word in a collection of documents. This is a standard benchmark application to test performance of different parallel tools in Big Data domain. In **Figure 14** the results of Apache Hadoop versus MapReduce on UPC for WordCount application is presented.

```
void * map (string filename)
{
char * file_data;
file_data = read_file_contents (filename);
Vector tokens;
vector_init(&tokens);
Tokenize (file_data,&tokens);
for (int i = 0;i < tokens.size;i++)
{collect (vector_get (&tokens,i),1);}
free(file_data);}
```

```
void reduce (string key,shared [] vector_sh *values)
{
int i;
int cnt = 0;
for (i = 0;i < values- > size;i++)
{int v = vector_get_shared_copy (values,i);
cnt + =v;}
reduce_collect (key,cnt);}
```

The presented MapReduce framework was developed using UPC parallel programming language which belongs to a family of PGAS languages. The overall obtained performance



```
void * map (string filename)
{
char * file_data;
file_data – read_file_contents (filename);
Vector tokens;
vector_init(&tokens);
Tokenize (file data,&tokens);
for (int i – 0;i<tokens.size;i++)
{collect (vector_get (&tokens,i),1);}
free(file_data);}
```

```
void reduce (string key,shared [] vector sh
*values)
{
  int i;
    int cnt – 0;
    for (i =0;i<values->size;i++)
    {   int v = vector get shared copy
(values,i);
          cnt +=v;   }
    reduce_collect (key,cnt);}
```

Figure 13. Implementation of map and reduce functions for WordCount application.

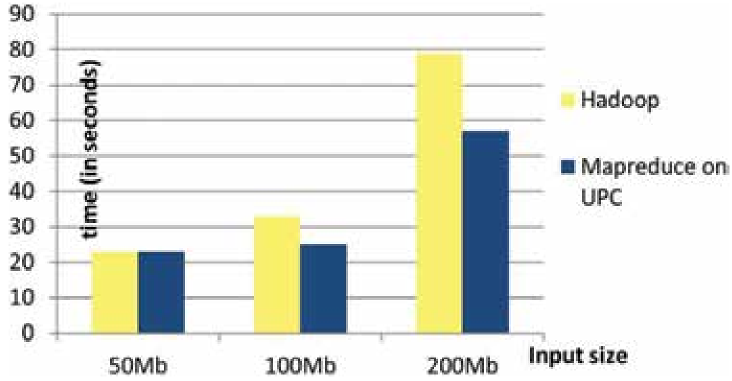


Figure 14. Apache Hadoop versus MapReduce on UPC.

and programmability benefits allow efficiently using this system for MapReduce based data processing tasks.

4. Conclusion

This chapter discusses high-performance computational and information technologies for numerical models and data processing. As a numerical model the oil recovery problem was considered. New fragmented algorithm was proposed, the algorithm for high-performance visualization and the algorithm on mobile platforms to solve this problem. Study of efficiency of applied algorithm implementations show that LuNA system appears to be less efficient than manual MPI implementation which justifies further development of LuNA functionality considering simplicity of software development with given system.

The described system contains the specialized visualization module implemented using Vulkan API. Given technology provides high-performance capabilities which were demonstrated using common desktop PC on a generated dataset.

The textual data processing problems as n-gram extraction and data clustering were also studied. In order to deal with computational complexity we had to adopt and implement parallelization patterns. Additionally, a new implementation of MapReduce framework was presented based on UPC language which provides functionality of combined MapReduce and partitioned global address space parallel programming models in a single execution environment which can be conveniently used in many complex workflows of data processing.

Author details

Darkhan Akhmed-Zaki, Madina Mansurova*, Timur Imankulov, Danil Lebedev, Olzhas Turar, Beimbet Daribayev, Sanzhar Aubakirov, Aday Shomanov and Kanat Aidarov

*Address all correspondence to: mansurova.madina@gmail.com

Al-Farabi Kazakh National University, Almaty, Kazakhstan

References

[1] Lake LW. Enhanced Oil Recovery. New Jersey: Prentice Hall Inc; 1989

[2] Sorbie KS. Polymer Improved Oil Recovery. Boca Raton: CRC Press; 1991

[3] Malyshkin V, Perepelkin V. Optimization methods of parallel execution of numerical programs in the LuNA fragmented programming system. The Journal of Supercomputing, Springer. 2012;**61**(1):235-248

[4] Vulkan. Industry Froged. Available from: https://www.khronos.org/vulkan/ [Accessed: 2017–05-01]

[5] BOINC - Open-Source Software for Volunteer Computing and Grid Computing. 2017. Available from: http://boinc.berkeley.edu/ [Accessed: 2017-05-01]

[6] Zhao D. Fast filter bank convolution for three-dimensional wavelet transform by shared memory on mobile GPU computing. The Journal of Supercomputing. 2015;**71**(9):3440-3455

[7] Montella R, Giunta G, Laccetti G, Lapegna M, Palmieri P, Ferraro C, Pelliccia V, Hong C, Spence I, Nikolopoulos D. On the virtualization of CUDA based GPU remoting on ARM and X86 machines in the GVirtuS framework. International Journal of Parallel Programming. 2017;**45**(5):1142-1163

[8] Lewis DD. Feature selection and feature extraction for text categorization. In: Proceedings of the Workshop on Speech and Natural Language. Harriman, New York: Association for Computational Linguistics; 1992. pp. 212-217

[9] Mikolov T, et al. editors. CoRR. 2013. arXiv: 1301.3781. Available from: http://arxiv.org/abs/1301.3781 [Accessed: 2017-05-01]

[10] Joulin A, et al. editors. Bag of Tricks for Efficient Text Classification. CoRR. 2016. arXiv: 1607.01759. Available from: http://arxiv.org/abs/1607.01759. [Accessed: 2017-05-01]

[11] Mikolov T, et al. editors. Distributed Representations of Words and Phrases and their Cmpositionality. CoRR. 2013. arXiv: 1310.4546. [Accessed: 2017-05-01]

[12] Whitley D, Sutton AM. Genetic algorithms. A survey of models and methods. In: Handbook of Natural Computing. Springer Berlin Heidelberg; 2012. pp. 637-671

[13] Bandyopadhyay S, Maulik U. Genetic clustering for automatic evolution of clusters and application to image classification. Pattern Recognition. 2002;**35**:1197-1208

[14] Gajawada S., D Toshniwal, N Patil, K Garg. Optimal clustering method based on genetic algorithm. In: Proceedings of the International Conference on Soft Computing for Problem Solving. December 20–22; 2011. pp. 295-303

[15] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. Communications of the ACM. 2008;**51**(1):107-113

[16] Dean J, Ghemawat S. Mapreduce: A flexible data processing tool. Communications of the ACM. 2010;**53**(1):72-77

[17] Bu Y, Howe B, Balazinska M, Ernst MD. The HaLoop approach to large-scale iterative data analysis. VLDB Journal. 2012;**21**(2):169-190

[18] Ekanayake J, Li H, Zhang B, Gunarathne T, Bae S-H, Qiu J, Fox G. Twister: A runtime for iterative MapReduce. In: Proceedings of the HPDC 2010. Chicago, IL, USA: ACM; 2010. pp. 810-818

[19] Zaharia M, Chowdhury M, Franklin MJ, Shenker S, Stoica I. Spark: Cluster computing with working sets. In: Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing. Boston, MA, USA: ACM; 2010. pp. 10-10

[20] Talbot JM, Yoo R, Kozyrakis C. Phoenix++: modular MapReduce for shared-memory systems. In: Proceedings of the Second International Workshop on MapReduce and its Applications. San Jose, California, USA: ACM; 2011. pp. 9-16

[21] Teijeiro C, Taboada GL, Tourino J, Doallo R. Design and implementation of MapReduce using the PGAS programming model with UPC. In: Proceedings of the 2011 IEEE 17th International Conference on Parallel and Distributed Systems (ICPADS '11). Tainan, Taiwan, 2011. pp. 196-203

[22] Dong H, Zhou S, Grove D. X10-enabled MapReduce. In: Proceedings of the Fourth Conference on Partitioned Global Address Space Programming Model (PGAS '10). New York, USA: ACM; 2010. p. 6

[23] Babalyan GA, Levy BI, Tumasyan AB, Khalimov EM. Oilfield development using surfactants. Moscow: Nedra; 1983. p. 98 (In Russian)

[24] Danaev N, Akhmed-Zaki D, Mukhambetzhanov S, Imankulov T. Mathematical modelling of oil recovery by polymer/surfactant flooding. Communications in Computer and Information Science. 2015:1-15

[25] Samarskii AA. Numerical methods. Moscow: Nauka; 1989. p. 432 (In Russian)

[26] Douglas J, Rachford HH. On the numerical solution of heat conduction problems in two and three space variables. Transactions of the American Mathematical Society. 1956;**82**(2): 421-439

[27] Samarskii AA. Theory of difference schemes: a tutorial. Moscow: Nauka; 1977. p. 656 (In Russian)

[28] Web site of the Interagency Supercomputer Center of the Russian Academy of Sciences. Available from: http://www.jscc.ru/scomputers.html [Accessed: 22.09.2017]

[29] Malyshkin VE, Perepelkin VA. LuNA fragmented programming system, main functions and peculiarities of run-time subsystem. In: Proceedings of the 11-th Conference on Parallel Computing Technologies, LNCS 6873. 2011. pp. 53–61

[30] AMD Mantle. Available from: http://www.amd.com/en-us/innovations/software-technologies/technologies-gaming/mantle [Accessed: 22.09.2017]

[31] DirectX 12. Available from: https://blogs.msdn.microsoft.com/directx/2014/08/12/directx-12-high-performance-and-high-power-savings/ [Accessed: 22.09.2017]

[32] Akhmed-Zaki DZ, Imankulov TS, Matkerim B, Daribayev BS, Aidarov KA, Turar ON. Large-scale simulation of oil recovery by surfactant-polymer flooding. Eurasian Journal of Mathematical and Computer Applications. 2016;**4**(1):12-31

[33] Akhmed-Zaki DZ, Daribayev BS, Imankulov TS, Turar ON. High-performance computing of oil recovery problem on a mobile platform using CUDA technology. Eurasian Journal of Mathematical and Computer Applications. 2017;**5**(2):4-13

[34] Cook Sh. CUDA Programming. A Developer's Guide to Parallel Computing with GPUs. Morgan Kaufmann: 2012. 600 p.

[35] NVIDIA Tegra X1 the new level of mobile performance. Available from: www.nvidia.com/object/tegra-x1-processor.html [Accessed: 22.09.2017]

[36] Brants T, Popat AC, Xu P, Och FJ, Dean J. Large Language Models in Machine Translation. In: Proceedings of the JCSSE; June 2007; Prague. pp. 858–867

[37] Berberich K, Bedathur S. Computing n-gram statistics in MapReduce. In: Proceedings of the 16th International Conference on Extending Database Technology (EDBT '13); 18–22 March 2013; Genoa. New York: ACM. pp. 101-112

[38] Aubakirov S, Trigo P, Ahmed-Zaki D. Comparison of distributed computing approaches to complexity of n-gram extraction. In: Proceedings of DATA 2016: 5th International Conference on Data Management Technologies and Applications. Lisbon: SCITEPRESS; 24-26 July. pp. 25-30

[39] Andrews BP, Binu A. Perusal on Hadoop small file problem. IJCSEITR. 2013;**3**(4):221-226

[40] Barakhnin VB, Nekhaeva VA, Fedotov AM. On the statement of the similarity measure for the clustering of text documents. Bulletin of Novosibirsk State University Series: Information Technology. 2008;**6**(1):3-9 (in Russian)

[41] Mansurova M, Barakhnin V, Aubakirov S, Khibatkhanuly E, Musina A. Parallel text document clustering based on genetic algorithm. In: Proceedings of the International Conference Mathematical and Information Technologies (MIT-2016); 28 August – 5 September 2016, Vrnjacka Banja. p. 218–232

[42] Shomanov AS, Akhmed-Zaki DZ, Mansurova ME. PGAS Approach to Implement Mapreduce Framework Based on UPC Language. In: Malyshkin V, editor. Parallel Computing Technologies. PaCT 2017. Lecture Notes in Computer Science. Vol. 10421. Cham: Springer. pp. 133-137

Chapter 2

The Eight Epochs of Math as Regards Past and Future Matrix Computations

Frank Uhlig

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.73329

Abstract

This survey paper gives a personal assessment of epoch-making advances in matrix computations, from antiquity and with an eye toward tomorrow. It traces the development of number systems and elementary algebra and the uses of Gaussian elimination methods from around 2000 BC on to current real-time neural network computations to solve time-varying matrix equations. The paper includes relevant advances from China from the third century AD on and from India and Persia in the ninth and later centuries. Then it discusses the conceptual genesis of vectors and matrices in Central Europe and in Japan in the fourteenth through seventeenth centuries AD, followed by the 150 year cul-de-sac of polynomial root finder research for matrix eigenvalues, as well as the superbly useful matrix iterative methods and Francis' matrix eigenvalue algorithm from the last century. Finally, we explain the recent use of initial value problem solvers and high-order 1-step ahead discretization formulas to master time-varying linear and nonlinear matrix equations via Zhang neural networks. This paper ends with a short outlook upon new hardware schemes with multilevel processors that go beyond the 0–1 base 2 framework which all of our past and current electronic computers have been using.

Math subject classifications: 01A15, 01A67, 65-03, 65F99, 65Q10

Keywords: math history, math computations, matrix, matrix computations, Zhang neural network, time-varying model, time-varying computations, 1-step ahead discretization formulas, time-varying equations, eigenvalues, computer hardware, numerical analysis

1. Introduction

In this paper we try to outline the epoch making achievements and transformations that have occurred over time for computations and more specifically for matrix computations. We will trace how our linear algebraic concepts and matrix computations have progressed from the beginning of recorded time until today and how they will likely progress into the future. We take this limited tack simply because in modern times, matrices have become the elemental and universal tools for most any computation.

This evolution of our matrix methods will be described in broad strokes. My main emphasis is to trace the mathematical genesis of matrices and their uses and to learn how the modern matrix concept has evolved in the past and how it is evolving. I am not interested in matrix theory by itself, but rather in matrix computations, i.e., how matrix concepts and algorithms have been developed from approximately 3000 BC to today, and even tomorrow.

This paper describes eight noticeably separate epochs that are distinguished from each other by the introduction of evolutionary new concepts and subsequent radically new computational methods. Following the historical trail through six historically established epochs, we will then look into the present and the near future.

What drives us to conceptualize and compute differently now, and what is leading us into the seventh and possibly eighth epoch? When and how will we likely compute in the future?

I am not a math historian, I have never taught a class in math history. Instead, throughout my academic career, I have worked with matrices: in matrix theory, in applications, and in numerical analysis. I like to construct efficient new algorithms that solve matrix equations. The idea for this paper is in part due to my listening by chance to a very short English broadcast from Egyptian radio on short wave some 40 years ago in the 1970s. It described an Egyptian papyrus from around 2000 BC that dealt with solving linear equations by row reduction and zeroing out coefficients in systems of linear equations, i.e., by what we now call "Gaussian elimination." When I heard this as a young PhD, I was fascinated and wrote the station for more information. They never answered, and when I was in Cairo many years later, the Egyptian Museum personnel could not help me either with locating the source.

Thus, I became aware that Carl Friedrich Gauß did not invent what we now call by his name, but who did?

For many decades, this snippet of math history just lingered in my mind until a year ago when I was sent a book on Zhang neural network (ZNN) methods for solving time-varying linear and nonlinear equations and was asked to review it. The ZNN methods were—to me and my understandings of numerics then—so other-worldly and brilliant that I began to think of the incredible leaps and "bounces" that math computations have gone through over the eons, from era to era. I eventually began to detect seven or eight computational sea changes, what I call "epochs," in our ability to compute with matrices, and that is my topic.

2. A short history of matrix computations

Nobody knows how numbers and number systems came about, just like nobody knows "who invented the wheel." I will start with a few historical facts about number systems and how they developed and were used across the globe in antiquity.

2.1. Early number systems

Humankind's first developments of number systems were very diverse and geographically widely dispersed, yet rather slow. The first circle cipher for zero occurred in Babylonia around 2500 BC or 4500 years ago. A continent or two removed, the Mayans used the same concept and circle zero symbol from around 40 BC. In India, it was recognized during the seventh century. But zero only became recognized as a "number to compute with" like all the others in the 9th century in Central India. Our decimal system builds on the ten numbers 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9. The decimal positional system came from China via the Indus valley, and it started to be used in Persia in the ninth century. It was combined with or derived from a Hindu number system of the same time period.

In fact Westerners call the current decimal number symbols wrongfully "Arabic," but most Westerners (and I) cannot read the license plates on cars in Egypt since the Arabic world does not use our Persian/Hindu numbers in writing but its own script using Arabic letters to designate numbers. Should we call our "western" numbers "Farsi" or "Hindu" instead?

Various bases have been used for numbering. There have been base 2, base 8, base 10, base 12, base 16, base 60, and base 200 number systems and possibly more at some time somewhere throughout human history. Counting and simple computations started with notched sticks for record keeping and with the invention of sand or wax tablets and then the abacus. These simple tools were developed a little bit differently and independently in many parts of the globe.

2.2. Antiquity: first epoch

Around 2200 to 1600 BC, Sumerian, Babylonian, and Egyptian land survey computations became mathematized in order to mark and allot land after the yearly Euphrates, Tigris, and Nile floods. That lead to linear equations in 2, 3, or 4 variables and subsequent methods to solve them that amounted to what we now call row reduction or Gaussian elimination.

Mathematical computations did not advance much during the Greek times as Greek mathematicians were mainly interested in mathematical theory and in establishing the concept of a formal proof, as well as elementary number theory of which the Euclidean algorithm is still used today.

Neither did the complicated Roman numerals lend themselves to easy computations, and no further computational advances happened there.

2.3. Early mathematical arts in China, India, and the Near East: second epoch

(Based in part on a lecture at Hong Kong University in 2017, given by Xiangyu Zhou, Chinese Academy of Sciences, for Chinese sources, and on Indian and Arabic sources from elsewhere).

In prehistoric and historic times (1600 BC–1400 AD), knot and rod calculus were prevalent in China. They were based on a decimal positional system, so-called rod numerals. These comprised the most advanced number system of the time, and it was used for several millennia before being adopted and expanded in Persia and India in the ninth century AD and later on adopted in Central Europe.

The *Mathematical Classic of Sunzi* by Suanjing (from the third to the fifth century) gives a detailed description of the arithmetic rules for counting rods. In the Indus valley, clay tablets covered with sand were used for mathematical computations several millennia ago. Bhaskara (600–680 AD) in India was the first one to write numbers in the Hindu positional decimal system which used the circle for zero. In 629 AD he approximated the sine function by rational expressions while commenting on Aryabhatta's (476–550 AD) book *Aryabhatiyabhisya* from 499 AD. An Indian contemporary of Bhaskara, Brahmagupta (598–665 AD), was the first one to establish the rules that govern computing with zero. Brahmagupta texts were written in Sanskrit verse that used the Sanskrit word for "eyes" to denote 2, "senses" for the number 5, etc. This was common in Indian mathematics and science writings at the time. The earliest record of multiplication and division algorithms using the Hindu numerals 1 through 9 and 0 was in writings by Al Khwarizmi 780–850 AD, a Persian mathematician employed in Bagdad. His *The Book of Manipulation and Restoration* established the golden rule of Algebra that an equation remains true if one subtracts the same quantity from both sides. He also wrote down multiplication and division rules that are identical to those of Suanjing from the third to fifth century in China. To Suanjing we also owe the Chinese remainder theorem. Finally, the advanced Hindu-Arabic decimal number system was introduced into the west by Leonardo Fibonacci (1175–1250 AD) of Pisa in his *Liber Abaci* or *The Book of Calculations* (1202),

Applied and numerical computations were driving much of Chinese mathematics. Wang Xiaotang (580–640 AD), for example, tried to find the roots of cubic polynomials that appeared in civil engineering and water conservation problems. In the *Mathematical Treatise in Nine Sections* of 1247, Qin Jiushao (1202–1261) developed the "Qin Jiushao method" which is now commonly called the "Horner-Ruffini scheme" for computing with and finding roots of polynomials iteratively. William George Horner [1] and Paolo Ruffini (1804–1807–1813) reinvented the Qin Jiushao method unknowingly 600 years later.

Chinese rod calculus was the method of choice for computing in China until the abacus took over during the Ming dynasty (1388–1644). Cheng Dawei (1355–1606) is the author of the first "numerical analysis" book titled *The General Source of Computational Methods* published in 1592. It describes methods to add, subtract, multiply, and divide on an abacus. The abacus itself was invented in various incarnations at various times and in several locations of the globe. It essentially combines several decimal rods on one board with beads on strings.

Chinese mathematicians from the third century BC onward to the tenth century AD brought us the *The Nine Chapters on the Mathematical Arts* that uses the numbers 1 through 9. This book was later disseminated further to the west and to India and Persia as described above. In Chapter 7, determinants first appeared conceptually, while Chapter 8 abstracts the concept of linear equations to represent them by matrix-like coefficient tableaux. These "matrix equations" were solved in China, again by "Gaussian elimination," 1500 years before Gauß' birth and 1800 years after the middle-eastern seasonal flood prone countries had first used the Gaussian algorithm around 1800 BC. Gauß himself described the method as the "common method of elimination" in his papers, and mathematicians then attached his name to it as an honor.

2.4. The genesis of vectors and matrices: third epoch

To advance matrix computations further, there was a need to conceptualize coordinates and vectors in space.

In the fourteenth century AD, Nicole Oresme developed a system of orthogonal coordinates for describing Euclidean space. This idea was taken up by René Descartes in the seventeenth century and is familiar to all of us now under the concept of Cartesian coordinates. Thereby, the world became ready for matrices and matrix computations in their own right.

In 1683 Gottfried Leibnitz in Germany and Seki Kowa in Japan both unbeknownst to each other reinvented the concept of a matrix as a rectangular array of coefficients for studying linear equations. Leibnitz also used and suggested row elimination to simplify and find their solutions. These efforts enabled Gauß to repeat what the Egyptians had done four millennia earlier: he was asked to survey the lands of his ruler, the Archduke George Augustus of Hanover, and measure the size of this kingdom inside Germany in the early 1800s. Beginning in the 1820s, Gauß, as Professor of Geodesy (and not of Mathematics) in Göttingen, would measure the angles and distances between many of the highest points there, such as the Brocken; the Inselsberg, 104 km apart; and the hills around Göttingen, and later he expanded the surveys all the way to the North Sea coast. He and his assistants did this multiple times, preferably when the weather was clear. Thereby, they set up systems of linear equations with generally more equations than unknown due to repeated measurements on different days.

To solve these overdetermined and naturally "unsolvable" systems $Ax = b$, Gauß devised the normal equation $A^T Ax = A^T b$ (1823) [2] and solved them approximately. But the normal equations method eventually turned out to be numerically unsound. It took over a century to find out why, the reason being that condition numbers multiply (see Olga Taussky [3]).

2.5. Eigenvalues and the characteristic polynomial: fourth epoch

As differential operators and matrices were beginning to be investigated and dealt with by the early 1800s, their connections and similarities were slowly recognized in the mathematics world.

The replication of certain functions $f \neq 0$ by a given differential operator $\mathscr{A}$ was noticed first and became the subject of studies. What were the functions f for which $\mathscr{A} f = \alpha f$ for some scalar α? How could they be found from $\mathscr{A}$, and what about α?

In 1829, Augustin Cauchy [4, p. 175] began to view the erstwhile "eigenvalue equation" $\mathscr{A} f = \alpha f$ as a "null space equation," namely, $\mathscr{A} f - \alpha f = 0$ or $(\mathscr{A} - \alpha \ id) f = 0$ for the identity operator with $id \ f = f$ for all f. Complete knowledge of the eigenvalues α and eigenfunctions f of a differential operator $\mathscr{A}$ allowed for a simple sum representations of the general solution of the linear differential equation described by $\mathscr{A}$. Thus, Cauchy's "null space equation" became essential for determining the behavior of systems governed by linear differential equations.

Cauchy's knowledge of and interest in determinants (think of the Cauchy-Binet theorem) then led him to define the "characteristic polynomial" of a square matrix A in 1839 [5, p. 827] as $f_A(\alpha) = \det(A - \alpha I)$, and thereby he initiated renewed studies in polynomial root-finding algorithms in the hope of obtaining analogous diagonalization results for linear matrix times vector products. And the search for polynomial root finders was on. By modern-day hindsight, reducing the eigenvalue problem from an n^2 data problem of the entries of a matrix $A_{n,n}$ to one of the $n+1$ coefficients of its characteristic polynomial is data compression, and therefore it was doomed to fail. But that remained unrealized by the mathematics community for more than 100 years.

James Sylvester finally gave the tableau concept of matrices its name "matrix" in 1848 or 1850. And after roughly two decades *1829* → *1839* → *1848/50*, the first century of matrix theory or theoretical linear algebra had begun.

2.6. Back to matrix computations

Cauchy's idea led mathematicians to try and compute the characteristic polynomials of matrices and find their roots in order to understand the eigen-behavior of matrices. We still teach many concepts and lessons today that are based on the "characteristic polynomial" $f_A(x)$ of a matrix A. Why, we should ask ourselves. Because unfortunately studying "characteristic polynomials" in place of matrices has turned out be a costly dead end for computational and applied mathematics: in the century and a half that followed Cauchy's work, more than 4000 papers on computing the roots of polynomials were published, together with 200 to 300 books on the subject, bringing us many algorithms, all of which failed more often than not. Many illustrious careers and schools of mathematics were founded based on this unfortunate and ever elusive goal.

During the same period, two-dimensional (2D) hand-cranked computing machines were invented and built to effect long number multiplications and divisions. First by Charles Babbage, then as commercial geared adding machines that were still being used in office work well into the 1960s. These worked as two-dimensional abaci of sorts. But eventually digital (at first punch card fed) computers became the tools of our computational trade in science, in engineering, in business, in GPS, in Google, in social media, in large data, in automation, etc.

But how could we or would we find matrix eigenvalues accurately? A turnaround, a new method, a new computational epoch was needed. From where, by whom, and how?

2.7. Iterative matrix algorithms: fifth epoch

To move us forward, it appears that matrix methods themselves might have to be developed that would solve the matrix-intrinsic eigenvalue problem by themselves. But before that was possible, there were further unfortunate "detours."

Carl Friedrich Gauß—in his doctoral thesis in 1799 [6]—had disproved all earlier attempts to establish the fundamental theorem of algebra, i.e., that all polynomials over the real numbers can be factored into as many real or complex conjugate factors as their degree says. His thesis then included the first complete and correct proof of the fundamental theorem of algebra.

In 1824 Niels Abel [7] showed that the roots of some fifth-degree polynomials cannot be found by using radical expressions of their coefficients; Gauß never opened or read the submitted paper and thus in fact rejected it knowingly on the grounds that God would not have complicated the World thus ... for us. Abel published his result privately, a broken man. Évariste Galois [8, 9] extended Abel's result in 1830 by giving group theoretic conditions for polynomials to be solvable by radicals; the extended paper (introducing Galois theory) was originally rejected and appeared only posthumously in 1846 [10].

Cautioned by these "rejected" inconvenient results, the polynomial approach to matrix eigenvalue computations could have been shunned by clearer heads early on, but the "dead end" determinants and characteristic polynomial roots road was taken instead for more than a century. Note that Cauchy's matrix result and most other fundamental matrix results from the nineteenth century were formulated in terms of determinants and only in the mid-twentieth century did the term "matrix" appear in matrix theoretical article and book titles.

A matrix-based approach to the eigenvalue problem nowadays starts from the simple fact that for any n by n matrix A and any n vector b, the sequence of vector iterates $b, Ab, A^2b, \ldots, A^kb, \ldots, A^nb$ contains $n + 1$ vectors in n-space which makes these vectors linearly dependent. Their linear dependency then leads to an nth-degree polynomial $p_b(A)$ that sends b to zero. The vanishing polynomial for any b turns out to always be a factor of the characteristic polynomial of A and it can be found by Gaussian elimination rather than using determinants.

The same idea shows that vector iteration converges for every starting vector $b \neq 0$ and any given square matrix A and this has led engineers in the early twentieth century to construct iterative matrix algorithms that could solve linear equations and the matrix eigenvalue problem. Iterative matrix algorithms actually do go back further to the Jacobi method (1839, 1845) [11, 12], the so-called Gauß-Seidel method, invented by Seidel alone (1874) [13], and various SOR methods that are designed to solve linear systems iteratively. The latter generally use matrix splittings of A rather than vector iteration. For further thoughts on early iterative matrix methods, refer to Michele Benzi [14].

Alexei Krylov [15] introduced and studied the vector iteration subspaces $\text{span}\left\{b, Ab, \ldots, A^kb\right\}$ in their own right. Following his ideas, large sparse matrix systems are nowadays treated iteratively in so-called Krylov-based methods, both to solve linear equations and to find matrix eigenvalues. Standard widely used Krylov-type iterative matrix algorithms carry the names of steepest descent and conjugate gradient by Hestenes and Stiefel [16], Arnoldi [17], Lanczos [18]. Others are called GMRES, BICGSTABLE, QMR, ADI, etc. Most Krylov-type methods are matrix and problem specific, and they are now mostly used for huge sparse and structured matrices where direct or semi-direct methods cannot be employed due to their high computational and storage costs. Krylov methods generally rely on preconditioner M for a linear system $Ax = b$ that shifts the spectrum of $M^{-1}A$ for faster convergence, and they thrive on incomplete matrix splittings, etc. Typically, they give only partial results. Who would need or want to know all million eigenvalues of a million by million matrix model. Krylov methods can be tuned to give information where it is needed for the underlying system.

2.8. Francis algorithm and matrix eigenvalues: sixth epoch

The Second World War (WW2) and post-Second World War periods were filled with innovations.

The atomic era had begun, as well as rocket science; commercial air flight became popular; and digital computers were being developed, first as valve machines and later transistorized. Supersonic speeds were realized, computer science was developed, etc. But there were many crashes and disasters with the new technologies: commercial aircraft (Super-Constellation, Convair, etc.) and military ones (Starfighter, etc.) would crash weekly around the globe; and newly built suspension bridges would collapse in strong winds.

The crux of the matter was that while matrix models of the underlying mechanical systems could readily be made using the laws of physics and mechanics, no one could reliably compute their eigenvalues. Engineers could not test their designs for eigenmodes in the right half plane! And Krylov methods were unfortunately not sufficient for testing for eigenvalues in a half plane.

If a matrix model of a mechanical or electrical or other structure, circuit, et cetera has right half-plane eigenvalues λ, then—upon proper excitation—there would be an eigen-component of the ever-increasing form $e^{\lambda t} \to \infty$ as $t \to \infty$ that resonates and self-amplifies inside the structure itself. This then leads to ever-increasing destructive vibrations and ultimate failure. The aircraft "flutter problem" was discovered during the Second World War. In England during WW2, Gershgorin circles that contain all of a system's eigenvalues were drawn out in the complex plane by rather primitive valve computers and checked to ascertain system stability.

The general matrix eigenvalue problem was finally solved independently and similarly by John Francis in London and by Vera Kublanovskaya in Russia nearly simultaneously around 1960. Francis' (or the QR) algorithm [19, 20] is based on Alston Householder's idea to try and solve matrix problems by matrix factorizations. Francis' method is an orthogonal subspace projection method and it works differently than the Krylov-based methods which solve a given matrix eigenvalue problem by projecting onto a Krylov subspace that is derived from and suitable for A.

A "divide and conquer" matrix factorization strategy was first employed by Heinz Rutishauser (1955, 1958) [21, 22] in his LR matrix eigenvalue algorithm: if one can factor $A = LR$ into the product of a lower and an upper triangular matrix L and R as $A = LR$ and if L is invertible, then for the reverse order product $A_1 = RL$ we have $A_1 = L^{-1}AL$ since $R = L^{-1}A$. If A_1 again allows an LR factorization $A_1 = L_1R_1$ with L_1 nonsingular, then by reverse order multiplication we obtain

$$A_2 = L_1^{-1}A_1L_1 = L_1^{-1}L^{-1}ALL_1$$

and so for the sequence of likewise constructed matrices A_i for $i = 3, \ldots$ if the respective LR factorizations are possible at each stage i. In this case the iterates A_i clearly remain similar to the original matrix A, and thus the iterates all have the same eigenvalues as A. Note, however, that if, for example, the (1,1) entry $A(1,1)$ is zero in A, then there exists no un-pivoted LR factorization for A, and the method breaks down since pivoting is a one-sided matrix process and not a similarity. Therefore, Rutishauser's method is only applicable to a limited set of matrices A for which every LR iterate A_i allows an un-pivoted LR factorization. Rutishauser had noted that if LR factorizations are possible for all iterates A_i, and the A_i becomes nearly upper triangular for large i with A's eigenvalues on the diagonal. (Very loosely said.)

John Francis was very interested in the flutter problem at the time when, by chance, someone dropped Rutishauser's 1958 LR paper [22] on his desk at the CRDC in London. *(In my interview with John Francis in 2009* [23], *he did not know who that might have been.)* Francis was aware through contacts with Jim Wilkinson of the backward stability of algorithms that involve orthogonal matrices Q. So rather than using Gaussian elimination matrices L, Francis experimented with orthogonal $A = QR$ factorizations. At roughly the same time, Vera Kublanovskaya worked on an LQ factorization of A as $A = LQ$ and subsequent reverse order multiplications [24] in Leningrad, Russia. Her LQ algorithm would also compute the eigenvalues of matrices reliably.

Rutishauser had observed convergence speedup for his LR method when replacing A by $A - \alpha I$, i.e., shifting. Hence Francis experimented with shifts for QR and then established the "implicit Q theorem" [20] in order to circumvent computing eigenvalues of real matrices over the complex numbers. Implicit shifts also avoid rounding errors that would be introduced by explicit shifts. Francis' second paper (1962) [20] also contains a fully computed flutter matrix problem of size 24 by 24. The eigenvalues of such "large" problems had never before been computed successfully.

Francis' implicit Q theorem then allowed Gene Golub and Velvel Kahan [25] to compute singular values of large matrices for the first time, and this application later spawned the original Google search engine and brought us—in a way—into the Internet age.

In 2002 the multishift QR algorithm was developed by Karen Braman et al. [26, 27]. It relies on subspace iteration, extends Francis' QR, and combines it with Krylov like methods. This extension allows us today to compute the complete eigenvalue and singular-value structure of dense matrices of sizes up to 10,000 by 10,000 economically on laptops.

What is being missed today computationally? What epoch(s) might come next? Why and how?

2.9. Two new epochs ahead: the seventh and eighth epochs (yet to come)

Two new epoch generating impulses have become visible on the matrix computational horizon of today:

A. One expands our computational abilities from static problem-solving algorithms to real-time methods for time-varying problems.

B. The other involves computer hardware advances.

2.9.1. Time-varying problems and real-time solvers: seventh epoch

Our current best numerical codes can solve static problem very well; that is what they are designed for.

As we begin to rely more and more on time-dependent sensoring and on robotic manufacture, we need to learn how to solve our erstwhile static equations but now in real time and with time-varying coefficients and preferably accurately as well. It seems quite alluring to try and solve a time-varying problem by using the static time-dependent inputs at each instance statically. But such a naive solution cannot suffice since at the next time step, whose "solution" has just been computed "statically", the problem parameters have already changed and thus our "static" solution solves a completely different problem, which—unfortunately—has little value. *If any at all.*

2.9.2. Computer hardware: eighth epoch

Since the earliest electronic computing devices of the 1940s, all our computers have worked as giant and embellished Turing machines with logic gates, switches, and memory that rely only on two numerical states: 0 and 1 or on or off. Hence, all our computer data is stored and manipulated as sequences of 0 and 1.

Lately our computing ability has come up against the limits of storing and working with data and processors that can only deal with zeros and ones. Our processing speeds have not advanced significantly over the last couple of years; we are still stuck with 3–4 GHz processors. To alleviate this bottleneck, chip makers have created multiprocessor chips, and software firms have introduced better and quicker software and operating systems, but the basic processor speed has not budged much.

At this time computer scientist and manufacturers are trying to overcome this 0–1 bottleneck by replacing our 0–1 processors, chips, memories, and transistors by improved transistors and chips that can store and process multistates, such as 0-1-2-3-4 or 0-1-2-3-4-5-6-7-8 or even higher-numbered data representations. This could lead us to another computing sea change bringing us into a new computational epoch via hardware. And, further out on the horizon lies the possibility of having infinitely many quantum states based computers.

3. On neural network methods: seventh epoch (already under way)

The last century brought us valuable tools to solve most static problems that involve matrices.

Our current numerical matrix tools can solve static linear equations and matrix equations such as Sylvester or Lyapunov equations, as well as eigenvalue problems, and generalizations of all of these, both of the dense or structured and of the solvable or unsolvable kinds. Likewise, we can solve static optimization problems of all sizes and for nearly all structured matrices and thereby solve most if not all static applications.

But what can we do with such problems when the entries are time-varying and the problem parameters change over time?

In numerical computations, there has always been a see-saw between models that resulted in derivative-inspired differential equations and in linear algebra based matrix equations. Their respective computational advantages differ from problem to problem. Neural networks (NN) are an amalgam of matrix methods and differential methods and use a mixture of both. NN methods are designed to solve time-varying dynamical systems. Numerical methods for time-varying dynamical systems first came about in the 1950s and subsequently have gained strength in the 1980s and 1990s and beyond (see the introduction in Getz and Marsden [28], for example). There are essentially three ways to go about solving the dynamical systems via differential equations: homotopy methods, gradient methods, and ZNN neural network methods introduced by Yunong Zhang et al. [29]. To solve a time-varying equation $f(X(t)) = G(t)$, the ZNN method starts from the error equation $E(t) = f(X(t)) - G(t)$ and stipulates exponential decay of the error function $E(t)$ to zero by trying to solve the differential equation:

$$\dot{E}(t) = -\lambda E(t) \tag{1}$$

for a positive decay constant λ. Ideally, the error differential equation (1) of a given problem can be discretized using high-order and convergent 1-step ahead difference formulas for the unknown. Their aim is to predict or simulate the solution at time t_{k+1} in real time from earlier event instants t_j with $j \leq k$ with a high degree of accuracy and low storage cost. The ZNN method will be explained below for three standard time-varying problems.

3.1. A neural network approach to solve time-varying linear equations: $\mathbf{A(t)x(t)=b(t)}$

Here $A(t)$ is a nonsingular time-varying n by n matrix and $b(t) \in \mathbb{R}^n$ is a time-varying vector, respectively. Clearly, the unknown solution $x(t)$ of the associated linear equation $A(t)x(t) = b(t)$ will be time-dependent as well.

The first paper on Zhang neural networks (ZNN) was written by Yunong Zhang et al. [29]. Today, there are well over 300 papers, mostly in engineering journals that deal with time-varying applications of the ZNN method, either in hardware chip design for specialized computational tasks as part of a plant or machine or for time-varying simulation problems in computer algorithms and codes. Unfortunately, the ZNN method and the ideas behind ZNN are hardly known today among numerical analysis experts. The method itself starts with using Suanjing's and Al Khwarizmi's ancient rule for reducing equations which first appeared 1 1/2 millennia ago. Recall that this simple rule was also employed by Cauchy [4] to transform the static matrix eigenvalue problem from $Ax = \lambda x$ to $Ax - \lambda x = 0$ and finally to det $(A - \lambda I) = 0$. For the time-varying linear equation problem, Zhang's neural network method starts with

$$A(t)x(t) - b(t) = 0$$

and then works on the error function

$$E(t) = A(t)x(t) - b(t) : \mathbb{R} \rightarrow \mathbb{R}^n$$

and its time derivative $\dot{E}(t)$. Note that standard static methods would likely look at the error norm $||E||$ instead of the error function. Neural networks do not; they study the error function $E(t)$ instead. And they start with an implicit "ideal wish": *What could or should we wish for* E?

Time-varying computations would be near ideal if their error functions $E(t)$ were decaying exponentially fast as functions of t. This is impossible to achieve (or even ask for) with our best static equations and problem solvers of the twenty-first century. For static numerical matrix methods, backward stability is considered most desirable.

In Zhang NN methods stipulating that the error function $E(t)$ decreases exponentially fast over time to the zero function means that

$$\dot{E}(t) = -\lambda E(t) \text{ for some chosen number } \lambda \gg 0, \text{ the decay constant.}$$

Note that for the time-varying linear equation problem, we have

$$\dot{E}(t) = \dot{A}(t)x(t) + A(t)\dot{x}(t) - \dot{b}(t).$$

This leads to the following differential equation for the time-varying solution $x(t)$:

$$A(t)\dot{x}(t) = -\dot{A}(t)x(t) + \dot{b}(t) - \lambda(A(t)x(t) - b(t)). \tag{2}$$

And thus, we have transformed the time-varying linear equations problem into an initial value differential equation problem that needs to be solved for $t > 0$. This is where the different dynamical system methods split their ways. In Zhang neural networks, the continuous time differential equation (2) is then discretized for $0 < t_j < t_{end}$ and the ensuing derivatives are approximated by high-order difference quotients, with the one for the unknown $\dot{x}\left(t_j\right)$ being 1-step ahead and proven convergent, while the others such as for $\dot{A}\left(t_j\right)$ and $\dot{b}\left(t_j\right)$ in equation (2) above can be backward difference formulas. This process would then yield a way to generate $x\left(t_{j+1}\right)$ from earlier known data such as $x(t_k)$, $A(t_k)$, $\dot{A}(t_k)$ and $b(t_k)$ and $\dot{b}(t_k)$ for indices $k \leq j$. How to proceed in this problem from equation (2) with solving $A(t)x(t) = b(t)$ via ZNN methods is still an open problem, especially for large-scale sparse or structured time-varying linear equations since the matrix $A\left(t_j\right)$ encumbers the unknown $\dot{x}\left(t_j\right)$ on the left-hand side of equation (2) and there is no known 1-step ahead differentiation formula that can be used here.

The general idea that underlies ZNN methods for time-varying problems is to replace repeated matrix computations by solving linear differential equations and associated initial value problems for discrete instances $0 < t_j < t_{end}$ instead.

3.2. A Zhang neural network approach to find time-varying generalized matrix inverses $\mathbf{Y(t)}$ for time-varying full rank matrices $\mathbf{B(t)}$ so that $\mathbf{B(t)_{m,n}Y(t)_{n,m} = I_m}$

This section is based on joint work with Jian Li et al. [30].

3.2.1. Continuous problem formulation

For an m by n real time-varying matrix $B(t)$ of full rank m with $m \leq n$, we form the matrix-valued error function:

$$E(t) = B(t) - Y^{+}(t) \in \mathbb{R}^{m \times n} \tag{3}$$

where the upper + sign always means "generalized inverse." Then, we use the Zhang design formula:

$$\dot{E}(t) = -\lambda E(t) \tag{4}$$

with design parameter $\lambda > 0$. Based on [31, Lemma 3], we have

$$\dot{Y}^{+}(t) = -Y^{+}(t)\dot{Y}(t)Y^{+}(t). \tag{5}$$

And from equations (3) and (5), we obtain

$$\dot{E}(t) = \dot{B}(t) - \dot{Y}^{+}(t) = \dot{B}(t) + Y^{+}(t)\dot{Y}(t)Y^{+}(t). \tag{6}$$

Combining equations (4) and (6), we then get

$$\dot{B}(t) + Y^{+}(t)\dot{Y}(t)Y^{+}(t) = -\lambda\big(B(t) - Y^{+}(t)\big) \tag{7}$$

And, by right multiplying equation (7) with $Y(t)$, we have

$$\Big(\dot{B}(t) + Y^{+}(t)\dot{Y}(t)Y^{+}(t)\Big)Y(t) = -\lambda\big(B(t) - Y^{+}(t)\big)Y(t). \tag{8}$$

With $m \le n$, we have $Y^{+}_{m\times n}(t)Y_{n\times m}(t) = I_{m\times m}$, and thus

$$\dot{B}(t)Y(t) + Y^{+}(t)\dot{Y}(t) = -\lambda(B(t)Y(t) - I). \tag{9}$$

The solution of a generalized matrix inverse problem is not unique when $m < n$, and we only need to find a solution that satisfies equation (9). Consequently, the continuous model can be represented as

$$\dot{Y}(t) = -\lambda(Y(t)B(t)Y(t) - Y(t)) - Y(t)\dot{B}(t)Y(t) \tag{10}$$

which agrees completely with [28, formula (15), p. 317]. Substituting equation. (10) into equation. (9), we have

$$\dot{B}(t)Y(t) + Y^{+}(t)\big(-\lambda(Y(t)B(t)Y(t) - Y(t)) - Y(t)\dot{B}(t)Y(t)\big) = -\lambda(B(t)Y(t) - I), \tag{11}$$

which we rewrite as

$$\dot{B}(t)Y(t) + \big(-\lambda\big(Y^{+}(t)Y(t)B(t)Y(t) - Y^{+}(t)Y(t)\big) - Y^{+}(t)Y(t)\dot{B}(t)Y(t)\big) = -\lambda(B(t)Y(t) - I). \tag{12}$$

With $Y^{+}_{m\times n}(t)Y_{n\times m}(t) = I_{m\times m}$, we have

$$\dot{B}(t)Y(t) + \left(-\lambda(B(t)Y(t) - I) - \dot{B}(t)Y(t)\right) = -\lambda(B(t)Y(t) - I). \tag{13}$$

Thus model (10) satisfies model (9), and its solution solves the time-varying generalized matrix inverse problem.

3.2.2. Zhang neural network discretization

Given a sequence of rectangular matrices B_j at time instances $t_j \leq t_k$, we want to find the discrete time-varying generalized matrix inverse Y_{k+1} of B_{k+1} on each computational time interval $[k\tau, (k+1)\tau) \subseteq [0, t_f]$ so that

$$B_{k+1} - Y_{k+1}^{+} = 0. \tag{14}$$

Here $B_{k+1} = B(t_{k+1}) = B((k+1)\tau) \in \mathbb{R}^{m \times n}$ is a time-varying full rank equidistant matrix sequence, $m \leq n$, and $Y_{k+1} \in \mathbb{R}^{n \times m}$ is unknown. Y_{k+1} needs to be computed in real time for each time interval $[k\tau, (k+1)\tau) \subseteq [0, t_{end}]$. Here the matrix operator $^+$ denotes the generalized inverse of a matrix and $0 \in \mathbb{R}^{m \times n}$ is the zero matrix. Besides, $k = 0, 1, \cdots$ denotes the updating index, t_{end} denotes the task duration, and τ denotes the constant sampling gap of the time-varying matrix sequence B_{k+1}. For $m > n$, the procedure is similar.

Note that we must obtain each Y_{k+1} at or before time t_{k+1} for real-time calculations, while the actual value of B_{k+1} is unknown before t_{k+1}. Thus we cannot obtain the solution by calculating $Y_{k+1} = B_{k+1}^{+}$. To obtain Y_{k+1} in real time, we must develop a model based on the available information before t_{k+1} such as that in B_j, Y_j, and Y_{j-1} for $j \leq k$ instead of unknown information such as B_{k+1}.

To obtain a discrete time model that solves the original discrete time-varying generalized matrix inverse problem (14), we need to discretize the continuous model (10). First, we use the conventional 1-step forward Euler formula:

$$\dot{x}(t_k) \approx \frac{x(t_{k+1}) - x(t_k)}{\tau} \tag{15}$$

with truncation error of order $O(\tau)$. Based on formula (15), we approximate

$$\dot{Y}(t_k) = \frac{Y(t_{k+1}) - Y(t_k)}{\tau} \tag{16}$$

and use this equation to discretize the continuous model (10) as follows:

$$Y_{k+1} = -h(Y_k B_k Y_k - Y_k) - \tau Y_k \dot{B}_k Y_k + Y_k. \tag{17}$$

Here $h = \tau\lambda$. In most real-world applications, information of the first-order time derivatives, i.e., the value of $\dot{B}_k$, may not be explicitly known for the discrete time-varying generalized matrix inverse problem (14). If this is so, the value of $\dot{B}_k$ can be approximated by a backward finite difference formula. To assure the accuracy and simplicity of the discretized model, the truncation error of the backward finite difference formula for $\dot{B}_k$ should be near equal to that of the 1-step ahead finite difference formula that approximates $\dot{Y}_k$. Thus, $\dot{B}_k$ in equation (17) should best be approximated by Euler's backward finite difference formula:

$$\dot{b}_k \approx \frac{b_k - b_{k-1}}{\tau}, \tag{18}$$

because the truncation error order $O(\tau)$ of formula (18) equals that of formula (15). Thus, we approximately have

$$\dot{B}_k = \frac{B_k - B_{k-1}}{\tau}. \tag{19}$$

Then we combine equation (17) with equation (19) and the Euler discrete model becomes

$$Y_{k+1} = -h(Y_k B_k Y_k - Y_k) - Y_k(B_k - B_{k-1})Y_k + Y_k. \tag{20}$$

Note that the truncation error of the discrete model (20) is of order $\mathbf{O}(\tau^2)$ where the symbol $\mathbf{O}(\tau^2)$ denotes a matrix in which each entry is of order $O(\tau^2)$. This model uses only present or past information of B_k, B_{k-1}, and Y_k and solves for Y_{k+1}. Thus Y_{k+1} can be calculated during the time interval $[t_k, t_{k+1})$ and if Y_{k+1} can be computed quickly enough in real time it will be ready when time instant t_{k+1} arrives.

Higher-accuracy 1-step ahead formulas exist for discrete models, namely,

$$\dot{x}(t_k) \approx \frac{2x(t_{k+1}) - 3x(t_k) + 2x(t_{k-1}) - x(t_{k-2})}{2\tau} \tag{21}$$

and

$$\dot{x}(t_k) \approx \frac{6x(t_{k+1}) - 3x(t_k) - 2x(t_{k-1}) - x(t_{k-2})}{10\tau}. \tag{22}$$

Both have truncation errors of order $O(\tau^2)$. For simplicity we only consider formula (21) and call it the 4-IFD formula because four instances in time are used to approximate the first-order derivative of $x(t_k)$. When we employ the 4-IFD formula (21) inside our continuous model (10), we obtain

$$Y_{k+1} = -h(Y_k B_k Y_k - Y_k) - \tau Y_k \dot{B}_k Y_k + \frac{3}{2} Y_k - Y_{k-1} + \frac{1}{2} Y_{k-2}. \tag{23}$$

Next, we use the three-instant backward finite difference formula:

$$\dot{b}_k \approx \frac{3b_k - 4b_{k-1} + b_{k-2}}{2\tau} \tag{24}$$

with error order $O(\tau^2)$ to approximate the value of $\dot{B}_k$ in equation (23). Then the 4-IFD-type discretized model becomes

$$Y_{k+1} = -h(Y_kB_kY_k - Y_k) - Y_k\left(\frac{3}{2}B_k - 2B_{k-1} + \frac{1}{2}B_{k-2}\right)Y_k + \frac{3}{2}Y_k - Y_{k-1} + \frac{1}{2}Y_{k-2}. \tag{25}$$

Its truncation error is of order $\mathbf{O}(\tau^3)$. Similar to the Euler based discrete model (20), the 4-IFD-type discrete model uses only present and past information such as B_k, B_{k-1}, B_{k-2}, Y_k, Y_{k-1}, and Y_{k-2} to solve for Y_{k+1}. Thus it also satisfies the requirements for real-time computation.

3.2.3. *A five-instant finite difference formula*

Any usable finite difference formula for discretizing the continuous model (10) must satisfy several restrictions. It must be one step ahead for $\dot{x}$, i.e., approximate $\dot{x}(t_k)$ by using only $x(t_{k+1})$, $x(t_k)$, $x(t_{k-1})$ and possibly earlier x data, and it must be 0-stable and convergent. However, 1-step ahead finite difference formulas do not necessarily generate stable and convergent discrete models (see, e.g., [32, 33]).

Here is a new 1-step ahead finite difference formula with higher accuracy than the Euler and 4-IFD formulas. It will be used to generate a stable and convergent discrete model that finds time-varying generalized matrix inverses more accurately in real time.

Theorem 1 *The 5-IFD formula*

$$\dot{x}(t_k) \approx \frac{8x(t_{k+1}) + x(t_k) - 6x(t_{k-1}) - 5x(t_{k-2}) + 2x(t_{k-3})}{18\tau} \tag{26}$$

has truncation error order $O(\tau^3)$.

The proof relies on four Taylor expansions that use $x(t_{k+1})$ and $x(t_{k-1})$ through $x(t_{k-3})$ around $x(t_k)$ and clever linear combinations thereof.

The new 1-step ahead discretization formula (26) then leads to the five-instant discrete model:

$$\begin{aligned} Y_{k+1} = \quad & -\frac{9}{4}h(Y_kB_kY_k - Y_k) - \frac{9}{4}Y_k\left(\frac{11}{6}B_k - 3B_{k-1} + \frac{3}{2}B_{k-2} - \frac{1}{3}B_{k-3}\right)Y_k \\ & -\frac{1}{8}Y_k + \frac{3}{4}Y_{k-1} + \frac{5}{8}Y_{k-2} - \frac{1}{4}Y_{k-3}. \end{aligned} \tag{27}$$

which has a truncation error of order $\mathbf{O}(\tau^4)$.

Theorem 2 *The five-instant discrete model (27) is 0-stable.*

The multistep formula of the five-instant discrete model time-varying generalized matrix inverses has the characteristic polynomial:

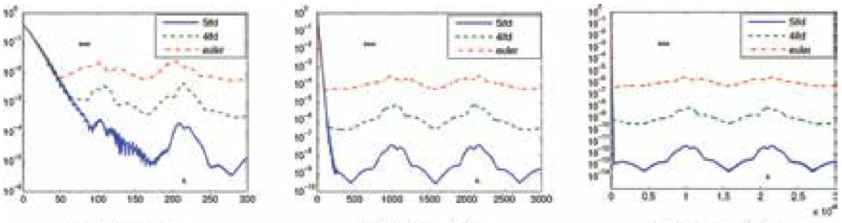
$$P_4(\theta) = \theta^4 + \frac{1}{8}\theta^3 - \frac{3}{4}\theta^2 - \frac{5}{8}\theta + \frac{1}{4} \tag{28}$$

with four distinct roots $\theta_{1,2} = -0.7160 \pm 0.5495i$, $\theta_3 = 0.3069$, and $\theta_4 = 1$ inside the complex unit circle, making this model 0-stable (**Figures 1** and **2**).

3.2.4. Numerical examples

Example 1 Consider the discrete time-varying generalized matrix inverse problem:

$$B_{k+1} - Y_{k+1}^{+} = 0, \text{with } B_k = \begin{bmatrix} \sin(0.5t_k) & \cos(0.1t_k) & -\sin(0.1t_k) \\ -\cos(0.1t_k) & \sin(0.1t_k) & \cos(0.1t_k) \end{bmatrix}. \tag{29}$$



(a) With $\tau = 0.1$ s

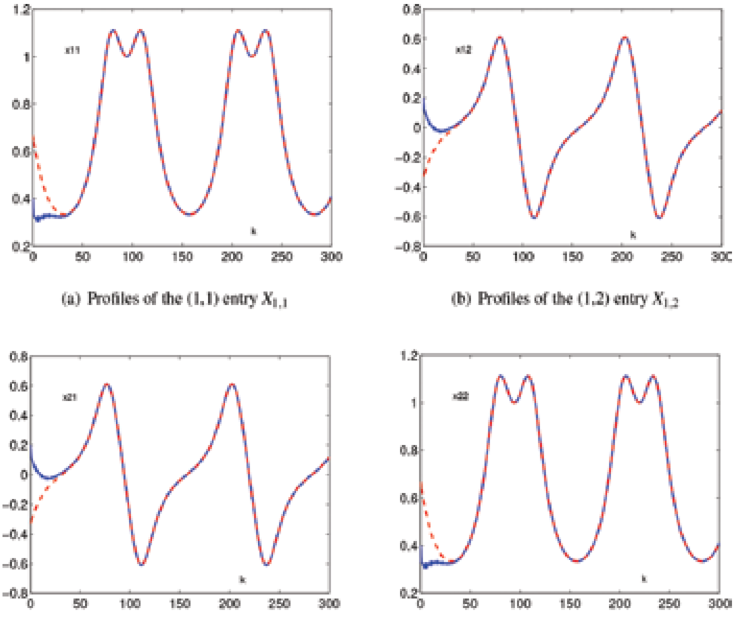
(b) With $\tau = 0.01$ s

(c) With $\tau = 0.001$ s

Figure 1. Typical residual errors generated by the five-instant, the four-instant, and the Euler formulas with different sampling gaps τ when solving the discrete time-varying generalized matrix inverse problem (29) for $t_{end} = 30$ s and $h = 0.1$.

Example 2 Here we consider the discrete time-varying matrix inversion problem:

$$A_{k+1}X_{k+1} = I \quad \text{with } A_k = \begin{bmatrix} \sin(0.5t_k) + 2 & \cos(0.5t_k) \\ \cos(0.5t_k) & \sin(0.5t_k) + 2 \end{bmatrix}. \tag{30}$$



(a) Profiles of the (1,1) entry $X_{1,1}$

(b) Profiles of the (1,2) entry $X_{1,2}$

(c) Profiles of the (2,1) entry $X_{2,1}$

(d) Profiles of the (2,2) entry $X_{2,2}$

Figure 2. Profiles of the four entries of the solution X when solving the discrete time-varying matrix inverse problem (30) with $\tau = 0.1$ s. Here the solid curves show the solution entries generated by the five-instant discrete model obtained from random starting values, and the dash-dotted curves depict the theoretical solutions.

3.3. A Zhang neural network approach for solving nonlinear convex optimization problems under time-varying linear constraints

This section is based on joint work with Jian Li et al. [34].

Problem formulation:

$$\begin{array}{lll} \text{find} & \min f(x(t),t) & \\ \text{such that} & A(t)x(t) = b(t) & \text{with } x(t) \in \mathbb{R}^n,\ \ b(t) \in \mathbb{R}^m \text{ and } A(t) \in \mathbb{R}^{m,n}. \end{array}$$

Building a continuous time model for the problem:

The Zhang neural network approach can be built on the Lagrange function:

$$L(x(t), l(t), t) = f(x(t), t) + l^T(t)(A(t)x(t) - b(t)),$$

where $l(t) = [l_1(t), \cdots, l_m(t)]^T \in \mathbb{R}^m$ is the Lagrange multiplier vector and $..^T$ denotes the transpose. Note that there will be no need to solve for the Lagrange functions $l(t)$ here. Set

$$y(t) = \left[x^T(t), l^T(t)\right]^T = \left[y_1(t), \cdots, y_n(t), y_{n+1}(t), \cdots, y_{n+m}(t)\right]^T \in \mathbb{R}^{n+m}$$

and

$$h(y(t), t) = \begin{bmatrix} \dfrac{\partial f(x(t), t)}{\partial x} + A^T(t) l(t) \\ A(t)x(t) - b(t) \end{bmatrix} = \begin{bmatrix} h_1(y(t), t) \\ \vdots \\ h_n(y(t), t) \\ h_{n+1}(y(t), t) \\ \vdots \\ h_{n+m}(y(t), t) \end{bmatrix} \in \mathbb{R}^{n+m}.$$

We transform the multiplier problem into an initial value DE problem instead. By stipulating exponential decay for $h(t)$, we obtain the model equation

$$\dot{y}(t) = -H^{-1}(y(t), t)(\lambda h y(t), t) + \dot{h}_t(y(t), t))$$

for the Jacobian matrix

$$H(y(t), t) = \begin{bmatrix} \dfrac{\partial f^2(x(t), t)}{\partial x\ \partial^T x} & A^T(t) \\ A(t) & 0 \end{bmatrix} \text{and } \dot{h}_t(y(t), t) = \frac{\partial h(y(t), t)}{\partial t} \ .$$

3.3.1. Discretizing the model and choosing suitable high-order finite difference formulas

To discretize the continuous model

$$\dot{y}(t) = -H^{-1}(y(t), t)\Big(\lambda h y(t), t) + \dot{h}_t(y(t), t)\Big)$$

we can use the forward Euler difference formula with truncation error order $O(\tau)$:

$$\dot{x}(t_k) = \frac{x(t_{k+1}) - x(t_k)}{\tau}$$

or the four-instant forward difference formula (4-IFD):

$$\dot{x}(t_k) = \frac{5x(t_{k+1}) - 3x(t_k) - x(t_{k-1}) - x(t_{k-2})}{8\tau}$$

with truncation error order $O(\tau^2)$. The Euler formula yields the discretized model:

$$y_{k+1} = -H^{-1}(y_k, t_k)\left(\kappa h(y_k, t_k) + \tau \dot{h}_t(y_k, t_k)\right) + y_k \text{ with } \kappa = \tau\lambda$$

while the 4-IFD formula results in

$$y_{k+1} = -\frac{8}{5}H^{-1}(y_k, t_k)\left(\kappa h(y_k, t_k) + \tau \dot{h}_t(y_k, t_k)\right) + \frac{3}{5}y_k + \frac{1}{5}y_{k-1} + \frac{1}{5}y_{k-2} + O(\tau^3).$$

Both discretization formulas are consistent and convergent. This can be proven via the roots of the associated characteristic polynomial. Its roots must lie in the complex unit circle and cannot be repeated on its boundary.

Since the value of $\dot{h}_t(y_k, t_k)$ may not be known explicitly, we may replace it by

$$\dot{h}_t(y_k, t_k) = \frac{3h(y_k, t_k) - 4h(y_k, t_{k-1}) + h(y_k, t_{k-2})}{2\tau}$$

which uses the three-point backward finite difference formula:

$$\dot{x}(t_k) = \frac{3x(t_k) - 4x(t_{k-1}) + x(t_{k-2})}{2\tau}$$

of order $O(\tau^2)$.

Then the discretized 4-IFD formula becomes more complicated but easier to implement:

$$\begin{aligned} y_{k+1} = -\frac{8}{5}H^{-1}(y_k, t_k)&\left(\left(\kappa + \frac{3}{2}\right)h(y_k, t_k) - 2h(y_k, t_{k-1}) + \frac{1}{2}h(y_k, t_{k-2})\right) \\ &+ \frac{3}{5}y_k + \frac{1}{5}y_{k-1} + \frac{1}{5}y_{k-2} \quad \text{of order} \quad O(\tau^3). \end{aligned}$$

To implement this formula, the inverse of the Jacobian matrix H can be computed at each time t_k in a fraction of the available real-time interval $[t_k, t_{k+1})$ by using the real-time inverse finding ZNN method from the previous subsection (Section 3.2).

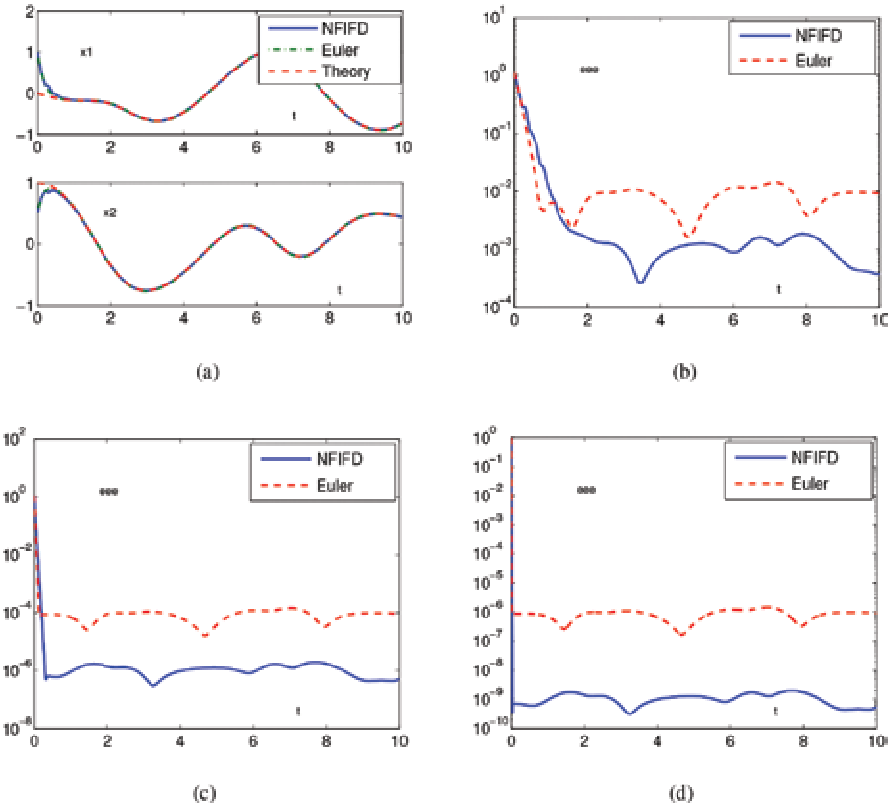
3.3.2. Numerical example and results:

As an example we solve the following convex nonlinear optimization problem with known theoretical solution numerically by using our ZNN method; for further details and applications see [34]:

$$\begin{array}{lll} \text{Find} & \min & (\cos(0.1t_{k+1})+2)x_1^2+(\cos(0.1t_{k+1})+2)x_2^2+2\sin(t_{k+1})x_1x_2+\sin(t_{k+1})x_1+\cos(t_{k+1})x_2 \\ & \text{so that} & \sin(0.2t_{k+1})x_1+\cos(0.2t_{k+1})x_2=\cos(t_{k+1}). \end{array}$$

The 4-IFD formula is a four-instant formula, while the Euler formula needs only two. Both discretization models work in real time, and both typically create the optimal solution in a fraction of a second with differing degrees of accuracy according to their orders.

The example below runs for 10 sec. The time-varying values for $f(x(t), t)$, $A(t)$, and $b(t)$ are given as functions and evaluated from their function formulations. In real-world applications, these values might be supplied by sensors during each time interval $t_i \leq t_{i+1}$, and the empirical values would be inserted into the difference formulas as they are evaluated by sensors in real time (**Figure 3**).



(a)

(b)

(c)

(d)

Figure 3. Solution states with $\tau = 0.1$ sec and solution errors generated by the 4-IFD based discretization model (of order $O(\tau^3)$) and the Euler based model (of order $O(\tau^2)$) with $\tau = 0.1$, 0.01, and 0.001 sec and $\lambda = 10$: (a) solution states with $\tau = 0.1$ sec, (b) solution errors with $\tau = 0.1$ sec, (c) solution errors with $\tau = 0.01$ sec, (d) solution errors with $\tau = 0.001$ sec.

4. On quantum and multistate computing: eight epoch (yet to start and come)

Quantum computing and multistate memory and computers with multistate processors will change the way we compute once they become available. They will require new operating systems and new software with new and yet-to-be-discovered algorithms. What will this new era entail? Nobody knows or can reliably predict.

I asked an "expert" on quantum computing 3 years ago as to when he expected to have a quantum computer at his disposal or on his desk. The answer was "Not in my lifetime, not in 20 years."

Currently, about a dozen or more research centers in Europe and South-East Asia are trying to build quantum computers based on the quantum superposition principle and quantum entanglement of elementary particles. They do so in a multitude of different ways. The envisioned benefit of these efforts would be to be able to compute superfast in parallel and in simulations to solve huge data problems quicker than ever before and to solve problems that are unassailable now with our current best supercomputer networks. All of the proposed quantum science techniques make use of superconducting circuits and particles. The aim is to build quantum computers in one or two decades with around 100 entangled quantum bits. Such a quantum computer would be bulky; it would need much supplementary equipment for cooling and so forth and could easily take up a whole floor of a building, just as the first German and British valve computers did in the 1940s. But it would surpass the computing capacity of all current supercomputers and desk and laptops on Earth combined. Currently, the largest working entangled quantum array contains fewer than 10 quantum bits. Access of a 100 bit quantum computer would probably be via the cloud and there would be no quantum computer laptops. Quantum computers may take another 10, 20, or 30 years to materialize.

How will they come about? Which yet unknown algorithms will they use? Who will invent them? Who code them?

If history can be a guide, John Francis and Vera Kublanovskaya were both working independently on circuit diagrams and logic gate designs for valve computers in England and in Russia at the time when they discovered QR (or LQ) in the late 1950s.

So, we possibly are looking for quantum computer hardware and software designers who know numerical analysis and algorithm development in or about the year 2040. In a similar fashion, Leibnitz and Seki formalized our now ubiquitous matrix concept independently but simultaneously in 1683, in Germany and in Japan.

Maybe it will take two again?

The references given below only go back to the year 1799.

Author details

Frank Uhlig

Address all correspondence to: uhligfd@auburn.edu

Department of Mathematics and Statistics, Auburn University, Alabama, United States

References

[1] Horner W. G. A new method of solving numerical equations of all orders, by continuous approximation. Philosophical Transactions. Royal Society of London. July 1819:308-335

[2] Gauß C. F. Private letter to Gerling. December 26, 1823. Available at: http://gdz.sub.uni-goettingen.de/en/dms/loader/img/?PID=PPN23601515X|LOG_0112physid=PHYS_0286. pp. 278-281

[3] Taussky O. Note on the condition of matrices. Mathematical Tables and Other Aids to Computation. 1950;**4**:111-112

[4] Cauchy A. L. Sur l'équation à l'aide de laquelle on determine les inégalités séculaires des mouvements des planètes. Exerc. de Math. 1829;**4**. also Oeuvres (2) 9, pp. 174-195

[5] Cauchy A. L. Mémoire sur l'intégration des équations linéaires. Comptes Rendus. 1839;**8**: 827-830, 845-865, 889-907, 931-937

[6] Gauß C. F. Demonstratio nova theorematis omnem functionem algebraicam rationalem integram unius variabilis in factores reales primi vel secundi gradus resolvi posse. PhD thesis: Universität Helmstedt; 1799, Werke III. pp. 1-30

[7] Abel N. H. Mémoire sur les équations algébriques où on démontre l'impossibilité de la résolution de l'équation générale du cinquième degré. Christiana (Kopenhagen): Groendahl; 1824. 7 p

[8] Galois É. Analyse d'un mémoire sur la résolution algébraique des équations. Bulletin des Sciences mathématiques, physiques et chimiques. 1830;**XIII**:271-272

[9] Galois É. Note sur la résolution des équations numériques. Bulletin des Sciences mathématiques, physiques et chimiques. 1830;**XIII**:413-414

[10] Galois É. Mémoire sur les conditions de résolubilité des équations par radicaux. Journal de mathématiques pures et appliquées, (published by Joseph Liouville). 1846;**XI**:417-433

[11] Jacobi C. G. Canon Arithmeticus. Berlin: Typis Academicis Berolini; 1839

[12] Jacobi C. G. Über eine neue Auflösungsart der bei der Methode der kleinsten Quadrate vorkommenden linearen Gleichungen. Astronomische Nachrichten. 1845;**22**:297-306. Reprinted in Gesammelte Werke, vol. III, pp. 469-478

[13] Ludwig von Seidel, Über ein Verfahren, die Gleichungen, auf welche die Methode der kleinsten Quadrate führt, sowie lineäre Gleichungen überhaupt, durch successive Annäherung aufzulösen, Lecture at the Bavarian Academy on February 7, 1874, Abhandlungen der Bayerischen Akademie der Wissenschaften II. Mathematisch-Physikalische Klasse. Cl. XI, Bd. III Abth, München. 1874. 28 pp

[14] Benzi M. The early history of matrix iterations: With focus on the Italian contribution. SIAM Conference lecture. October 2009. 36 pp. https://www.siam.org/meetings/la09/talks/benzi.pdf

[15] Krylov A. N. On the Numerical Solution of Equation by Which are Determined in Technical Problems the Frequencies of Small Vibrations of Material Systems. News of Academy of Sciences of the USSR), Otdel. mat. i estest. nauk. 1931;**VII**(4):491-539 (in Russian)

[16] Hestenes M. R., Stiefel E. Methods of conjugate gradients for solving linear systems. Journal of Research of the National Bureau of Standards. 1952;**49**(6):409-436

[17] Arnoldi W. E. The principle of minimized iterations in the solution of the matrix eigenvalue problem. Quarterly of Applied Mathematics. 1951;**9**:17-29

[18] Lanczos C. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. Journal of Research of the National Bureau of Standards. 1950;**45**:255-282

[19] Francis J. G. F. The QR transformation, I. The Computer Journal. 1961;**4**:265-271

[20] Francis J. G. F. The QR transformation, II. The Computer Journal. 1962;**4**:332-345

[21] Rutishauser H. Une methode pour la determination des valeurs propres d'une matrice. Comptes Rendus de l'Académie des Sciences. 1955;**240**:34-36

[22] Rutishauser H. Solution of eigenvalue problems with the LR-transformation. National Bureau of Standard: Applied Mathematics Series. 1958;**49**:47-81

[23] Golub G, Uhlig F. The QR algorithm: 50 years later; its genesis by John Francis and Vera Kublanovskaya, and subsequent developments. IMA Journal of Numerical Analysis. 2009;**29**:467-485

[24] Kublanovskaya V. N. On some algorithms for the solution of the complete eigenvalue problem. USSR Computational Mathematics and Mathematical Physics. 1963;**1**:pp. 637-657 (1963, received Feb 1961). Also published in: Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki [Journal of Computational Mathematics and Mathematical Physics]. 1 (1961). pp. 555-570

[25] Gene Golub W. Kahan, calculating the singular values and pseudo-inverse of a matrix. Journal of SIAM Numerical Analysis Series B. 1965;**2**:205-224

[26] Braman K, Byers R, Mathias R. The multishift QR algorithm. Part I: Maintaining well-focused shifts and level 3 performance. SIAM Journal on Matrix Analysis & Applications. 2006;**23**:929-947

[27] Braman K, Byers R, Mathias R. The multishift QR algorithm. Part II: Aggressive early deflation, SIAM Journal on Matrix Analysi & Application. 2006;**23**:948-973

[28] Getz N, Marsden J. E. Dynamical methods for polar decomposition and inversion of matrices. Linear Algebra and its Applications. 1997;**258**:311-343

[29] Zhang Y, Jiang D, Wang J. A recurrent neural network for solving Sylvester equation with time-varying coefficients. IEEE Transactions on Neural Networks. 2002;**13**:1053-1063

[30] Jian L, Mingzhi M, Frank U, Yunong Z. A 5-Instant finite difference formula to find discrete time-varying generalized matrix inverses, matrix inverses and scalar reciprocals, 17 pp, submitted

[31] Liao B, Zhang Y. From different ZFs to different ZNN models accelerated via li activation functions to finite-time convergence for time-varying matrix pseudoinversion. Neurocomputing. 2014;**133**:512-522

[32] Zhang Y, Jin L, Guo D, Yin Y, Chou Y. Taylor-type 1-step-ahead numerical differentiation rule for first-order derivative approximation and ZNN discretization. Journal of Computational and Applied Mathematics. 2014;**273**:29-40

[33] Zhang Y, Chou Y, Chen J, Zhang Z, Xiao L. Presentation, error analysis and numerical experiments on a group of 1-step-ahead numerical differentiation formulas. Journal of Computational and Applied Mathematics. 2013;**239**:406-414

[34] Li J, Mao M, Uhlig F, Zhang Y. Z-type neural-dynamics for time-varying nonlinear optimization under a linear equality constraint with robot application. Journal of Computational and Applied Mathematics. 2018;**327**:155-166

Survey of Computational Methods for Inverse Problems

Sergey Voronin and Christophe Zaroli

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.73332

Abstract

Inverse problems occur in a wide range of scientific applications, such as in the fields of signal processing, medical imaging, or geophysics. This work aims to present to the field practitioners, in an accessible and concise way, several established and newer cutting-edge computational methods used in the field of inverse problems—and when and how these techniques should be employed.

Keywords: inverse problems, matrix factorizations, regularization, parameter estimation, model appraisal, seismic tomography

1. Introduction

In this work, we aim to survey several techniques useful to a practitioner in the field of inverse problems, where the solution to a vector of interest is given through a linear system $Ax = b$ or through a set of nonlinear equations $F(x) = 0$. In our presentation below, we review both classical results and newer approaches, which the reader may not be familiar with. In particular, this chapter offers entries on the following material:

- Matrix factorizations and sparse matrices
- Direct solves and pivoted factorizations
- Least squares problems and regularization
- Nonlinear least squares problems
- Low-rank matrix factorizations and randomized algorithms
- An introduction to Backus-Gilbert inversion

2. Notation

In this chapter, we use the norm $\|\cdot\|$ to refer to the spectral or operator norm, and $\|\cdot\|_p$ to refer to the ℓ_p norm. We make frequent use of the QR decomposition and the SVD (singular value decomposition). For any $M \times N$ matrix A and (ordered) subindex sets J_r and J_c, $A(J_r, J_c)$ denotes the submatrix of A obtained by extracting the rows and columns of A indexed by J_r and J_c, respectively; and $A(:, J_c)$ denotes the submatrix of A obtained by extracting the columns of A indexed by J_c. We will make use of the covariance and the variance matrix, which we define as follows in terms of the expected value:

$$\operatorname{cov}(x, y) = E[x - E[x])(y - E[y])^T] \text{ and } \operatorname{var}(x) = \operatorname{cov}(x, x).$$

By "Diag," we refer to a diagonal matrix with nonzeros only on the diagonal. We make use of the so-called GIID matrices. These are matrices with independent and identically distributed draws from the Gaussian distribution. In the Octave environment (which we frequently reference), these can be obtained with the "randn" command. We assume the use of real matrices, although most techniques we describe extend to the complex case.

3. Matrix factorizations and sparse matrices

Let A be an $M \times N$ matrix with real or complex entries, and set $r = \min(M, N)$. We will make use of the singular value decomposition (SVD) of a real matrix $A \in \mathbb{R}^{M\times N}$: if A is of rank r, then there exist $U \in \mathbb{R}^{m\times r}$, $V \in \mathbb{R}^{N\times r}$ and $\sum \in \mathbb{R}^{r\times r}$ such that

1. $U^T U = I$, $V^T V = I$,
2. $\sum = \operatorname{Diag}(\sigma_1, \sigma_2, \ldots, \sigma_r) \in \mathbb{R}^{r\times r}$ is a diagonal matrix with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, and
3. $A = U\Sigma V^T$.

This is known as the economic form of the SVD [1]. For $1 \leq i \leq \min\{M, N\}$, the i-th largest singular value of A is defined to be σ_i, with $\sigma_j = 0$ for $j = r + 1, \ldots, \min\{M, N\}$ whenever $r < \min\{M, N\}$. The generalized inverse of $A \in \mathbb{R}^{m\times N}$ with $SVD A = U\Sigma V^T$, is defined as $A^+ = V\Sigma^{-1}U^T$ (and $\sum^{-1} = \operatorname{Diag}(\sigma_1^{-1}, \sigma_2^{-1}, \ldots, \sigma_r^{-1}) \in \mathbb{R}^{r\times r}$). By a rank deficient matrix, we imply a nonlinear decay of singular values $\{\sigma_i\}$. In this case, the numerical rank of A may be smaller than the actual rank due to the use of finite precision arithmetic.

The (compact) QR-factorization of A takes the form

$$\underset{m\times n}{A} \quad \underset{n\times n}{P} \quad = \quad \underset{m\times r}{Q} \quad \underset{r\times n}{R}, \tag{1}$$

where P is a permutation matrix, Q has orthonormal columns, and R is upper triangular. The permutation matrix P can more efficiently be represented via a vector $J_c \in \mathbb{Z}_+^n$ of indices such that $P = I(:, J_c)$ where I is the $n \times n$ identity matrix. The factorization (1) can then be written as:

$$\begin{array}{ccccc} A(:,J_c) & = & Q & R \\ m \times n & & m \times r & r \times n \end{array} \tag{2}$$

Another commonly used decomposition is the pivoted LU:

$$\begin{array}{ccccc} A(:,J_c) & = & L & U. \\ m \times n & & m \times r & r \times n \end{array} \tag{3}$$

with L a lower triangular and U an upper triangular matrix. In the Octave environment, these decompositions can be constructed, respectively, via the commands $([Q, R, I] = qr(A, 0);$ $[L, U, I] = lu(A)$. The matrix P does not need to be explicitly formed. Instead, vector I gives the permutation information. The relation between the two in Octave is given by the command $\mathrm{P}(:, \mathrm{I}) = \mathrm{eye}\,(\mathrm{length}(\mathrm{I}))$.

Many matrices in applications turn out to be sparse. They can be stored more efficiently, without the need to store all $m \times n$ elements. The simplest sparse format is the so called coordinate sparse format, common to, e.g., the Octave environment. In this format, we store the integers row, column, and floating point value for each nonzero of the sparse matrix A: a set of triplets of the form (i, j, v). However, we do not need to store all the row and column indices of the nonzero elements. Below, we summarize the two commonly used sparse formats for an example matrix.

$$A = \begin{bmatrix} 0 & 6 & 3 & 0 \\ 1 & 0 & 8 & 0 \\ 7 & 0 & 0 & 2 \end{bmatrix}$$

The compressed column and row formats for this matrix are given by the vectors:

$$i_c = [1, 2, 0, 0, 1, 2], \quad p_c = [0, 2, 3, 5, 6], \quad d_c = [1, 7, 6, 3, 8, 2],$$

and

$$i_r = [2, 1, 0, 2, 3, 0], \quad p_r = [0, 2, 4, 6], \quad d_r = [3, 6, 1, 8, 2, 7].$$

In the compressed column format, the d_c array stores the nonzero elements, scanned row by row. The array i_c stores the row index of the corresponding data element, and the array p_c stores

```
function y = mat_mult (A, x)
  y = zeros (m, 1);
  for i = 1:m
    for j = pr (i): pr (i + 1)
      y (i) = y (i) + dr (j) * × (i r (j));
    end
  end
end
```

```
function y = mat_trans_mult (A, x)
  y = zeros (n, 1);
  for i = 1:m
    for j = pr (i): pr (i + 1)
      y (i r (j)) = y (i r (j)) + dr (j) * × (i);
    end
  end
end
```

the index of the start of each column in the data array d_c. Similarly, for the compressed row format, all the column indices of nonzeros are given, but the row information is compressed by giving in p_r, the index of the start of each row in d_r. Moreover, if needed, the three vectors for the sparse representation above can be further compressed, with, e.g., lossless compression techniques, such as arithmetic coding [2]. BLAS operations on sparse matrices can be performed directly using these storage formats. Below, we list the pseudocode for the operations $y_1 = Ax_1$ and $y_2 = A^T x_2$ for a $m \times n$ sparse matrix A stored with compressed row format.

4. Direct solves

Given a linear system $Ax = b$ with a square matrix A which is invertible $(\det(A) \neq 0)$, the solution x can be constructed through the inverse of A, built up using Gaussian elimination. For relatively small systems, the construction of such solutions is often desired over least squares formulations, when a solution is known to exist. Typically elimination is used to construct the factorization of A into a QR or LU decomposition. The construction of factorizations (QR, LU) with column pivoting can be applied to system solves involving rank deficient matrices. As an example, consider the pivoted QR factorization $AP = QR$. Here $AP = A(:; I)$, a rearrangement of the columns of A upon multiplication with permutation matrix P. Plugging into $Ax = b$ yields $QRP^T x = b \Rightarrow QRy = b \Rightarrow Ry = Q^T b$, which is an upper triangular system, and can be solved by back substitution. A simple permutation $P^T x = y \Rightarrow x = Py$ yields the solution x.

Similarly, suppose we have the pivoted LU factorization $AP = LU$. Then, plugging into $Ax = b$ yields $LUP^T x = b$. Next, set $z = UP^T x = Uy$ with $y = P^T x$. Then $Lz = b$ (which is a lower triangular system) can be solved by forward substitution for z, while $Uy = z$ can be solved by back substitution for y. Again applying a permutation matrix to $x = Py$ yields the result. Notice that multiplying by P can be done efficiently, simply be re-arranging the elements of y. The implementations of the back substitution and forward substitution algorithms are given below.

```
% Solve Lz = b

function z = fwd_sub (L, b)
  n = length (b);
  z = zeros (n, 1);
  for i = 1:n
    z (i) = (b (i) − L(i,:) * z) /L(i, i);
  end
end
```

```
% Solve Uy = z

function y = back_sub (U, z)
  n = length (z);
  y = zeros (n, 1);
  for i = n: −1:1
    y (i) = (z (i) − U(i,:) y) /U(i, i);
  end
end
```

5. Regularization

5.1. Least squares

Prior to discussing two-norm or what is more commonly known as Tikhonov regularization, we mention the least squares problem:

$$\overline{x}_{\text{lsq}} = \arg\min_x \|Ax - b\|_2^2 \tag{4}$$

This formulation arises due to the noise in the right hand side b, in which case it does not make sense to attempt to solve the system $Ax = b$ directly. Instead, if we have an estimate for the noise norm (that is, $b = \overline{b} + e$ with unknown noise vector e, but we can estimate $\|e\|_2$), then we could seek *a* solution x such that $\|Ax - b\|_2 \approx \|e\|_2$. Let us now look at the solution of (4) in more detail. As the minimization problem is convex and the functional quadratic, we obtain the minimum by setting the gradient of the functional to zero:

$$\nabla_x \|A_x - b\|_2^2 = 0 \Rightarrow A^T Ax = A^T b \tag{5}$$

A common choice of solution to the quadratic Eq. (5) would be directly through the generalized inverse:

$$x = \left(A^T A\right)^+ A^T b = \left(V\Sigma^{-2}V^T\right)V\Sigma U^T b = A^+ b, \tag{6}$$

because of all the solutions to $A^T Ax = A^T b$, $A^+ b$ has the smallest ℓ_2-norm: $A^T Ax = A^T b$ if and only if $x = A^+ b + d$ for some $d \in \ker\left(A^T A\right) = \text{range}\left(A^T A\right)^\perp = \text{range}\left(A^+\right)^\perp$, and

$$\left\|A^+ b + d\right\|^2 = \left\|A^+ b\right\|^2 + 2d^T\left(A^+ b\right) + \|d\|^2 = \left\|A^+ b\right\|^2 + \|d\|^2 \left\|A^+ b\right\|^2.$$

Typically, the least squares problem is solved by an iterative method such as conjugate gradients (CG) or related techniques such as LSQR. Whichever way the solution to the normal equations in (5) is obtained, it will be close A^+b and share its properties.

First, if A has small singular values, the norm of the solution $A^+ b = V\Sigma^{-1}U^T b$ will be very large because of the Σ^{-1} matrix. Another disadvantage of (4) is that the solution A^+b will be very sensitive to any noise in b or even in A (if any approximations in the calculations are used). Suppose the noise vector e behaves like white noise. Its different entries are uncorrelated, each having mean 0 and standard deviation ν. If, in addition, the elements of $\overline{b}$ and e are uncorrelated, we have:

$$\text{var}(e) = E\left[(e - E[e])(e - E[e])^T\right] = E\left[ee^T\right] = \nu^2 I,$$

and

$$\mathrm{var}(b) = E\Big[(b - E[b])(b - E[b])^T\Big] = E\big[ee^T\big] = \nu^2 I.$$

We may then estimate the norm of the variance of the solution vector:

$$\begin{aligned}&\mathrm{var}(x) = \mathrm{var}\big(A^+ b\big) = \mathrm{var}\big((V\Sigma^{-1}U^T)b\big) = \big(V\Sigma^{-1}U^T\big)\mathrm{var}(b)\Big(U\big(\Sigma^{-1}\big)^T V^T\Big) = \nu^2 V\Sigma^{-2}V^T \\ &\Rightarrow \|\mathrm{var}(x)\|_2 = \nu^2\big\|V\Sigma^{-2}V^T\big\|_2 = \frac{\nu^2}{\sigma^2_{\min}},\end{aligned}$$

where $\sigma_{\min}$ is the smallest magnitude singular value of A. We can clearly see that when A has an appreciable decay of singular values (such that $\sigma_{\min}$ is small relative to σ_1), the solution $x = A^+ b$ will be sensitive to data errors. For these reasons, adding additional terms (regularization) to the optimization problem is often necessary.

5.2. Tikhonov regularization

Having discussed the least squares approach we turn our attention to the simplest form of Tikhonov Regularization:

$$\bar{x}_{\text{tik}} = \arg\min_x \Big\{ \|Ax - b\|_2^2 + \lambda\|x\|_2^2 \Big\} \tag{7}$$

where $\lambda > 0$ is a scalar regularization parameter, which controls the tradeoff between the solution norm $\|x\|_2$ and the residual fit $\|Ax - b\|_2$. Since the functional in brackets is convex, we can get the solution by again setting the gradient to zero:

$$\nabla_x \Big\{ \|Ax - b\|_2^2 + \lambda\|x\|_2^2 \Big\} = 0 \Rightarrow 2A^T(A\bar{x} - b) + 2\lambda\bar{x} = 0 \Rightarrow \big(A^T A + \lambda I\big)\bar{x} = A^T b \tag{8}$$

If we plug in the SVD of $A = U\Sigma V^T$, we get:

$$\begin{aligned}\bar{x} &= \Big(\big(U\Sigma V^T\big)^T\big(U\Sigma V^T\big) + \lambda I\Big)^{-1} A^T b \\ &= \Big(V\big(\Sigma^T\Sigma + \lambda I\big)^{-1} V^T\Big) V\Sigma^T U^T b \\ &= V\big(\Sigma^T\Sigma + \lambda I\big)^{-1}\Sigma^T U^T b \\ &= V\,\mathrm{Diag}\left(\frac{\sigma_i}{\sigma_i^2 + \lambda}\right) U^T b = VDU^T b\end{aligned}$$

We see that the effect of the regularization is to filter the small singular values σ_i, by replacing each σ_i by $\frac{\sigma_i}{\sigma_i^2+\lambda}$, which prevents the singular values smaller than λ from dominating the solution. If we now compute the norm of the solution variance, we obtain:

$$\mathrm{var}(\overline{x}) = \mathrm{var}\left(VDU^T b\right) = \left(VDU^T\right)\mathrm{var}(b)\left(UDV^T\right) = v^2 VD^2V^T$$
$$\Rightarrow \quad \|\mathrm{var}(\overline{x})\|_2 = v^2\left\|VD^2V^T\right\|_2 = v^2\left\|D^2\right\|_2 \le \frac{v^2}{4\lambda}.$$

The result follows because the function $h(t) := \frac{t}{t^2+\lambda}$ satisfies $h'(t) = 0$ at $t = \pm\sqrt{\lambda}$ with $h\left(\sqrt{\lambda}\right) = \frac{1}{2\sqrt{\lambda}}$. Thus, the solution to the system from (8) (even if obtained from a CG type scheme after a finite number of iterations) relieves the problems due to small singular values and noise, which affect the solution from (5).

In practice, a slight generalization of (7) is performed by adding a regularization matrix L:

$$\overline{x}_{\mathrm{tikL}} = \arg\min_x \left\{\|Ax - b\|_2^2 + \lambda_1\|x\|_2^2 + \lambda_2\|Lx\|_2^2\right\} \tag{9}$$

Generally, L is some kind of sharpening (difference) operator. The penalization of the term $\|Lx\|$, thus results in smoothing. If the coordinate system x is one-dimensional, it could take the form:

$$L_1 = \begin{bmatrix} -1 & 1 & \dots & 0 & 0 \\ 0 & -1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -1 & 1 \end{bmatrix}$$

When the coordinate system corresponding to the model vector x is higher dimensional, L will be correspondingly more complicated and will generally have block structure. Note that the solution to (9) can be obtained through the linear equations:

$$\left(A^TA + \lambda_1 I + \lambda_2 L^T L\right)\overline{x} = A^T b \tag{10}$$

and can also be cast as an augmented least squares system and solved through its corresponding augmented normal equations:

$$\overline{x} = \arg\min_x \left\| \begin{bmatrix} A \\ \sqrt{\lambda_1} I \\ \sqrt{\lambda_2} L \end{bmatrix} x - \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix} \right\|_2^2 \Rightarrow \begin{bmatrix} A \\ \sqrt{\lambda_1} I \\ \sqrt{\lambda_2} L \end{bmatrix}^T \begin{bmatrix} A \\ \sqrt{\lambda_1} I \\ \sqrt{\lambda_2} L \end{bmatrix} \overline{x} = \begin{bmatrix} A \\ \sqrt{\lambda_1} I \\ \sqrt{\lambda_2} L \end{bmatrix}^T \begin{bmatrix} b \\ 0 \\ 0 \end{bmatrix}$$

This is an important point with regards to implementation, as it means that codes which solve the normal equations for standard least squares can be readily modified to solve (9). If L is a smoothing operator, the parameters λ_1 and λ_2 effect the degree of norm damping and model smoothing, respectively. Notice that increasing λ_2 from zero also changes the solution norm, so λ_1 may need to be altered to compensate.

5.3. Sparse regularization and generalized functional

The ℓ_2 penalty in (7) has the effect of penalizing the solution norm in a way which encourages all coefficients to be small (as λ is increased). For very large λ, only $x = 0$ would result in the required minimum but for modest values (below, e.g., $\|A^T b\|_\infty$), the effect would be to force all solution coefficients to have smaller magnitudes with increasing λ, but would not make any of them explicitly zero. In many applications, a sparse solution is sought (where, a significant percentage of the coefficients of x are zero). A so-called ℓ_0 measure is an indicator function for the number of nonzeros of x. This measure is not a norm, as it does not satisfy, e.g., the basic triangle inequality. Constraining the ℓ_0 measure (e.g., $\|Ax - b\| < \epsilon$, $\min \|x\|_0$), leads to a combinatorially difficult optimization problem.

A key insight of compressive sensing is that the minimization with respect to the ℓ_0 measure and the convex ℓ_1 norm (the sum of the absolute values of the components of a vector) produce the same result (the sparsest solution) under some strong conditions (i.e., RIP, restricted isometry property) on A [3]. In practice, a nonrandom A from a physical application would not satisfy the RIP conditions. On the other hand, the minimization with respect to the ℓ_1 norm (and the ℓ_p-norms for $0 < p < 2$, convex only for $p \geq 1$) produce sparse solutions (but not-necessarily the sparsest one at a given residual level). Sample illustrations for the 2d case are given in **Figure 2**, where we can observe that in two dimensions, the minimization of the ℓ_p norm for $p \leq 1$ results in one of the two components equal to zero. To account for the possibility of employing a sparse promoting penalty and also for more general treatment of the residual term, which we discuss more below, we will consider the two-parameter functional [4]:

$$F_{l,p}(x) = \|Ax - b\|_l^l + \lambda \|x\|_p^p = \sum_{i=1}^{m} \left| \sum_{j=1}^{m} A_{ij} x_j - b_i \right|^l + \lambda \sum_{i=1}^{n} |x_i|^p, \tag{11}$$

with $\tilde{F}_p = F_{2,p}$ (with $l = 2$ being the most-common residual-based penalty). For $p < 2$, the functional $\tilde{F}_p$ is not differentiable. This means that the minimum value cannot be obtained by setting the gradient equal to zero as for ℓ_2-based minimization. A particularly well-studied example is the ℓ_1 case, which is the closest convex problem to the ℓ_0 penalty. Convexity of $\tilde{F}_1(x)$

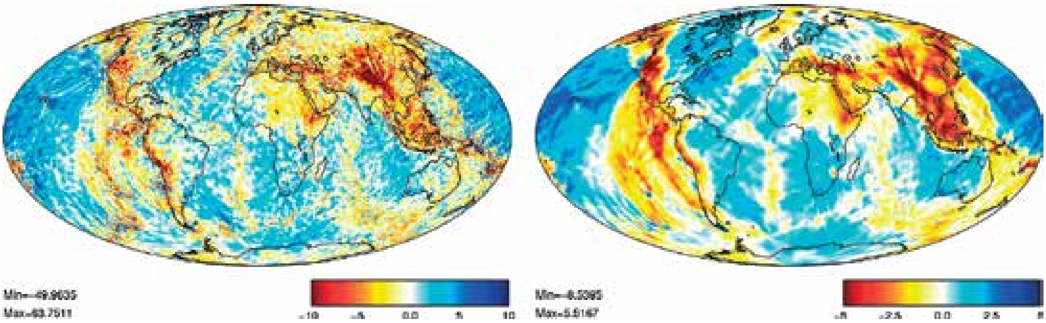


Figure 1. Comparison of geophysical models recovered via (10) with $\lambda_2 = 0$ and $\lambda_2 > 0$.

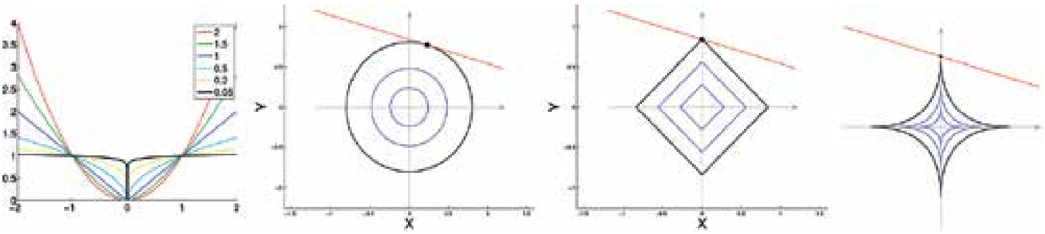


Figure 2. Illustration of family of functions $|y|^p$ for $p \in (0, 2)$ and sample solutions to $ax_1 + bx_2 = c$ subject to min $\|x\|_p$ for $p = 2, 1, 0.5$.

guarantees that any local minimizer is necessarily a global one. In this case, an algorithm can be constructed which decreases the functional value and tends to the (global) minimizer of $\tilde{F}_1(x)$. One such method is called the iterative soft thresholding algorithm (ISTA) and relies on the soft thresholding function $S_\tau(x)$, defined as:

$$(\mathbb{S}_\tau(x))_k = \text{sgn}(x_k)\max\{0, |x_k| - \tau\}, \quad \forall k = 1, \ldots, n, \forall x \in \mathbb{R}^n.$$

The benefit of this function is two-fold: it explicitly sets small components of x to zero (promoting sparsity) and is continuous (unlike the related hard thresholding function which simply zeros out all components smaller than τ in absolute value). The soft thresholding function satisfies a useful identity:

$$\mathbb{S}_\tau(b) = \arg\min_x \left\{ \|x - b\|_2^2 + 2\tau\|x\|_1 \right\},$$

which is utilized with a surrogate functional approach to construct the ISTA scheme:

$$x^{n+1} = \mathbb{S}_\tau\left(x^n + A^T b - A^T A x^n\right)$$

This algorithm converges to the ℓ_2 minimizer for any initial guess and with $\|A\|_2$ (the spectral norm of A) being less than 1 (which can be accomplished simply by rescaling). The spectral norm of a matrix A can easily be estimated using so called power iteration [1]. Let us assume that A is square. If it is not we can take the matrix $A^T A$ in it's place and take the square root of the eigenvalue found to be the estimate for the spectral norm. If we take a vector x^0 and write it as a linear combination of eigenvectors of A, then $x^0 = \alpha_1 v_1 + \ldots + \alpha_n v_n$. It follows that the iterative computation $x^m = Ax^{m-1}$ yields (plugging in $Av_k = \lambda_k v_k$):

$$\alpha_1 \lambda_1^m v_1 + \ldots + \alpha_n \lambda_n^m v_n = \lambda_1^m \left[\alpha_1 v_1 + \alpha_2 \left(\frac{\lambda_2}{\lambda_1} \right)^m v_2 + \ldots + \lambda_n \left(\frac{\lambda_n}{\lambda_1} \right)^m v_n \right] \Rightarrow \lim_{m \to \infty} \frac{x^m}{\lambda_1^m} = \alpha_1 v_1,$$

a scalar multiple of the dominant eigenvector. A simple computation yields the dominant eigenvalue. In practice, a much faster converging scheme called FISTA (Fast ISTA) [5] is utilized, which is a slight reformulation of ISTA applied to a linear combination of two previous iterates:

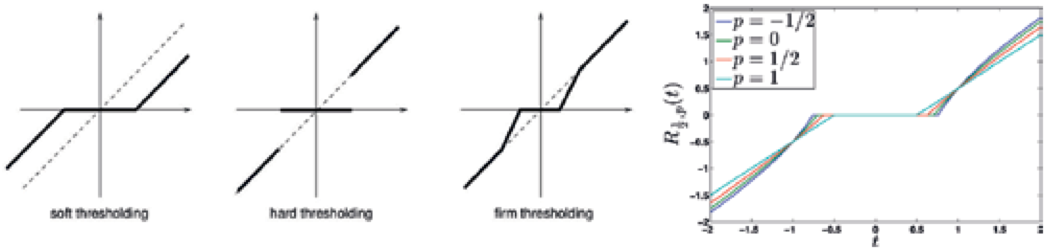


Figure 3. Illustrations of different thresholding functions [6, 7].

$$x^{n+1} = \mathbb{S}_\tau\left(y^n + A^Tb - A^TAy^n\right), \quad y^n = x^n + \frac{t_{n-1}-1}{t_n}\left(x^n - x^{n-1}\right), \quad t_{n+1} = \frac{1+\sqrt{1-4t_n^2}}{2}, \tag{12}$$

where $t_1 = 1$. This algorithm is simple to implement and readily adapts to parallel architectures. For challenging problems (such as when the decay of singular values of A is rapid), the thresholding function $\mathbb{S}_\tau$ can be slightly altered (see **Figure 3**), but the same iterative scheme (12) can be utilized. Two possible approaches are either to vary the thresholding, starting from soft thresholding and slowly approaching the (discontinuous) hard thresholding function, or to use a function which better mimics hard thresholding away from zero. The use of different thresholding functions alters the optimization problem being solved. Thresholding-based techniques are simple to implement but are not effective in all situations, particularly when only a few iterations are feasible (for example, when A is large). In this case, two interesting approaches are iteratively re-weighted least squares (IRLS) and convolution smoothing [4]. Both techniques replace the nonsmooth part of the functional (namely, the absolute value function $|x|$) by a smooth approximation. Moreover, both techniques have the particular advantage of being able to employ gradient-based methods (such as CG) at each iteration, considerably increasing the per-iteration performance. The IRLS approach is based on the approximation:

$$|x_k| = \frac{x_k^2}{|x_k|} = \frac{x_k^2}{\sqrt{x_k^2}} \approx \frac{x_k^2}{\sqrt{x_k^2+\epsilon^2}}$$

where in the rightmost term, a small $\epsilon \neq 0$ is used, to insure the denominator is finite, regardless of the value of x_k. The resulting algorithm [4] for the minimization of (11) can be written as:

$$\left(A^TR^nA + (D^n)^T(D^n)\right)x^{n+1} = A^TR^nb$$

with two diagonal iteration dependent matrices D_n and R_n. The diagonal matrix D^n has elements $\sqrt{\frac{1}{2}\lambda p w_k^n}$ and R^n has diagonal elements $l\left|r_i^n\right|^{l-2}$ (for i where $\left|r_i^n\right| < \epsilon$, we can set the entry to $l\epsilon^{l-2}$ with the choice of ϵ user controllable, tuned for a given application). Here, the residuals $r_i^n = (Ax^n - b)_i$ and the iteration dependent weights are given by:

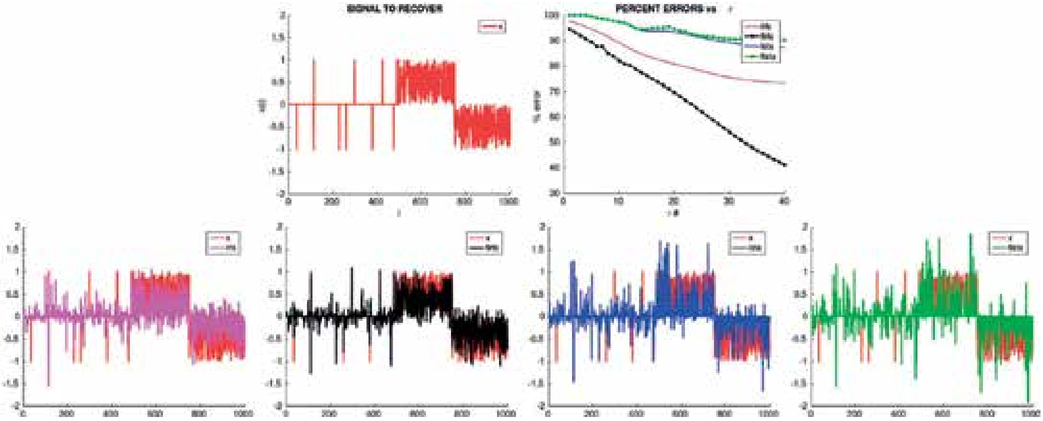


Figure 4. Illustration of half-sparse, half-dense signal recovery with different algorithms.

$$w_k^n = \frac{1}{\left[\left(x_k^n\right)^2 + \epsilon_n^2\right]^{\frac{2-p}{2}}}.$$

The diagonal matrices (or simply the vectors holding the diagonal elements) are updated at each iteration and the system in (6.10) can be solved approximately via a few iterations of CG- or LSQR-based algorithms. Another advantage of the IRLS approach is that the powers p can be made component dependent. This then allows for better inversion of partially sparse signals (if of course, the location of the sparse part can be estimated with some accuracy). An example is illustrated in **Figure 4** and further discussed in [8]. Another approach discussed in [4] is based on a smooth approximation of the absolute value function $f(t)$ = |t| obtained via convolution with a sequence of Gaussian kernels, which have approximately shrinking support. The resulting "conv-CG" method is suitable especially for rapid warm start acceleration.

5.4. Alternate penalty functions and regularization parameter selection

We mention here the classical image deconvolution problem. Given a blurring source g, such as a 2D Gaussian function, we can produce a blurry image s from an unaltered source f via convolution $s = f^* g + n$, where n is some additive noise component. For such situations, a TV (total variation) norm penalty is frequently used, for purposes of noise removal [9]. For a 2-D signal (such as an image), the TV penalty can be written as $V(s) = \sum_{i,j} \sqrt{\left|s_{i+1},j - s_{i,j}\right|^2 + \left|s_{i,j+1} - s_{i,j}\right|^2}$. Sometimes, the alternate approximation $\sum_{i,j} \left|s_{i+1,j} - s_{i,j}\right| + \left|s_{i,j+1} - s_{i,j}\right|$ is utilized. Various iterative schemes have been developed for such penalty functions [9].

Both the ℓ_2-based approaches and sparsity promoting regularization schemes (as well as TV-norm penalty functionals) utilize one or more regularization parameters. In the case of Tikhonov regularization with smoothing (as in (9)) more than one parameter is present. In this case, the second (smoothing) parameter can generally be set according to the desired smoothing effect, once the first parameter λ_1 is chosen (with a fixed value of λ_2) and then, λ_1 can be adjusted to achieve a desired solution norm. Thus, we focus here on techniques to adjust λ_1, which we simply refer to as λ.

The standard way to choose the parameter is to use the L-curve technique starting at a large λ (generally a value close to $\|A^T b\|_\infty$ is a good choice) and decreasing down in logarithmic fashion using the logarithmically spaced sequence:

$$S = \frac{\log(\lambda_{\max}) - \log(\lambda_{\min})}{N-1}; \quad \lambda_i = \exp(\log(\lambda_{\max}) - S(i-1)), \quad i = 1, \ldots, N.$$

The parameter N can vary by application but is typically in the range [5, 10]. Two typical Strategies for parameter selection are employed.

The first is based on a target residual value, typically determined by the estimate of the noise norm. At every λ after the initial value we reuse the previous solution as the initial guess for the CG scheme at the current λ. We can use the solution x_λ for which $\|Ax_\lambda - b\|$ is closest to the desired residual level (or refine further the solution at this λ with more CG iterations).

If however, the target residual norm is not available, other techniques must be used. We discuss a method using the so-called L-curve where for the norm damping problem (7), we plot a curve composed of points $(\log\|Ax_\lambda - b\|, \log\|x_\lambda\|)$ which we can obtain using the same continuation procedure previously discussed. The curve represents the tradeoff between the residual value $\|Ax_\lambda - b\|$ and the solution norm $\|x_\lambda\|$. In practice, neither of these quantities should dominate over the other. Hence, an established strategy is to look for the point of maximum curvature along the L-curve [11]. If we set:

$$\bar{\epsilon} = \log\|x_\lambda\|_p \quad \text{and} \quad \bar{\rho} = \log\|Ax_\lambda - b\|_l, \tag{13}$$

where x_λ is the solution of (11) at the particular value of λ. We can then compute the curvature by the formula:

$$\bar{c}_\lambda = 2\frac{\bar{\rho}'\bar{\epsilon}'' - \bar{\rho}''\bar{\epsilon}'}{\left((\bar{\rho}')^2 + (\bar{\epsilon}')^2\right)^{\frac{3}{2}}}, \tag{14}$$

where the derivative quantities can be approximated via finite differences. We illustrate various plots for a synthetic example in **Figure 5**. In the residual plot, the target residual is taken to be the magnitude of the noise vector norm. We can also see that the lowest percent error

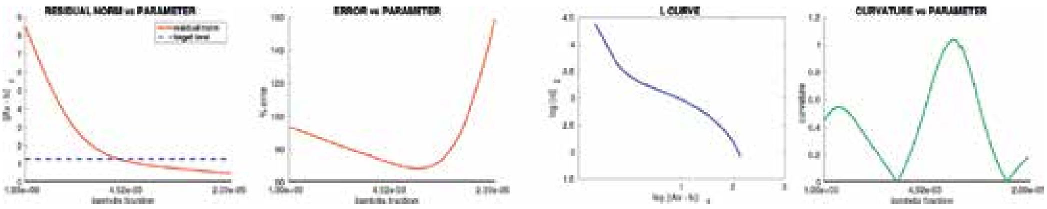


Figure 5. Regularization parameter picking. Set 1: Residuals and percent errors vs. λ fraction (fraction of $\|A^T b\|_\infty$). Set 2: L-curve and curvature curve as a function of λ fraction.

between $\overline{x}_\lambda$ and the true x occurs at a value of λ roughly corresponding to the highest curvature of the L-curve. In fact, for this example, the curvature method gives a better estimate of good λ than the residual curve technique.

6. Nonlinear least squares (NLS) problems

In many cases, the inverse problem may be posed in terms of a nonlinear function $F\ (x,\ t)$ with x a vector of variables, which may be time dependent with parameter t. We first describe, here, the popular Newton-Gauss method for NLS [1]. Let $g(x) = \frac{1}{2}\|r(x)\|^2$ with $r_i(x) = y_i - F(x, t_i)$. Then, the NLS problem takes the form: $\overline{x} = \arg\min_x g(x)$. Setting $\nabla g(x) = 0$, yields with Newton's method:

$$x^{n+1} = x^n - \left[\nabla^2 g(x^n)\right]^{-1} \nabla g(x^n)$$

Expanding the gradient and Hessian of g yields:

$$\begin{aligned}\nabla g(x) &= \sum_{i=1}^{m} r_i(x)\nabla r_i(x) = J^T r(x) \text{ where} J = J[r(x)]\\ \nabla^2 g(x) &= \sum_{i=1}^{m} \nabla r_i(x)\nabla r_i(x)^T + \sum_{i=1}^{m} r_i(x)\nabla^2 r_i(x) = J^T J + T(x) \approx J^T J.\end{aligned}$$

where $T(x) = \sum_{i=1}^{m} r_i(x)\nabla^2 r_i(x)$ and $J[r(x)]_{(i,:)} = \nabla r_i(x)^T = -\nabla F(x, t_i)^T$. The approximation $T(x) \approx 0$ is used in the expression for the Hessian in the Gauss-Newton method, yielding a simple iterative scheme:

$$x^{n+1} = x^n - \left[J_n^T J_n\right]^{-1} J_n^T r_n.$$

Unfortunately, this method is not stable and will typically not converge if initialized far away from a minimum solution [1]. Improvements include the introduction of a step size parameter:

$$\begin{aligned}&x^{n+1} = x^n - \alpha_n \left[J_n^T J_n\right]^{-1} J_n^T r_n\\ &\alpha_n = \arg\min_\alpha g(x^n - \alpha s^n) \quad \text{with} \quad J_n^T J_n s^n = J_n^T r_n.\end{aligned}$$

and of the use of a regularizer (e.g., Levenberg-Marquardt method [1]): where the system ${J_n}^T J_n y = {J_n}^T r_n$ is replaced by an ℓ_2-norm penalty regularized system, $\left({J_n}^T J_n + \lambda I\right)\tilde{y} = {J_n}^T r_n$.

7. Low-rank matrix factorizations

In many applications, there are large matrices with rapidly decaying singular values. In such cases, low-rank matrix approximations like the low-rank SVD are useful for compression, speed gains, and data analysis purposes. For $A \in \mathbb{R}^{m \times n}$, the low-rank SVD of rank k (with $k < \min(m, n)$) is the optimal matrix approximation of A in the spectral and Frobenius norms. Taking $p = \min(m, n)$, we define the low-rank SVD of rank k by A_k by taking into account only the first $k < p$ singular values and vectors: that is, with $U_k \in \mathbb{R}^{m \times k}$ consisting of the first k columns of $U, \Sigma_k = \mathrm{Diag}(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k \times k}$ consisting of k rows and columns of Σ, and $V_k \in \mathbb{R}^{n \times k}$ consisting of the first k columns of V:

$$A_k = \sum_{j=1}^{k} \sigma_j u_j v_j^T = U_k \sum\nolimits_k V_k^T, \tag{15}$$

$$U_k = [u_1\ u_2 \ldots u_k], \quad V_k = [v_1 v_2 \ldots v_k], \quad \text{and} \quad \sum = \begin{bmatrix} \sigma_1 & 0 & 0 & \cdots & 0 \\ 0 & \sigma_2 & 0 & \cdots & 0 \\ 0 & 0 & \sigma_3 & \cdots & 0 \\ \vdots & \vdots & \vdots & & 0 \\ 0 & 0 & 0 & \cdots & \sigma_k \end{bmatrix}$$

By the Eckart-Young theorem [12]:

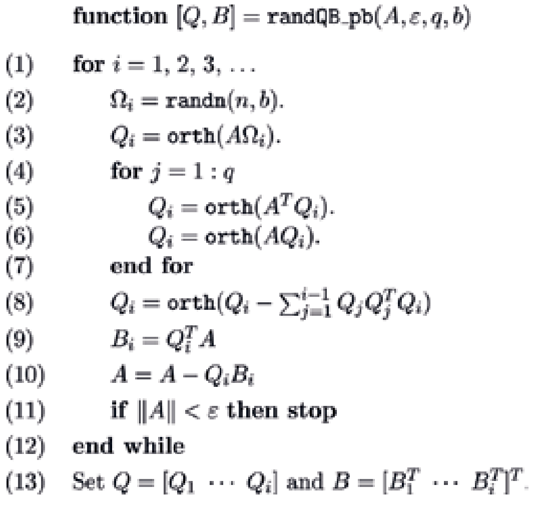
$$\|A - A_k\| = \sigma_{k+1},$$

when the error is measured in the spectral norm, and

$$\|A - A_k\|_F = \left(\sum_{j=k+1}^{p} \sigma_j^2 \right)^{1/2}$$

in the Frobenius norm. When $k \ll p$, the matrices U_k, Σ_k and V_k are significantly smaller (cost of storage of all nonzeros is $mk + nk + k$) than the corresponding full SVD matrices U, Σ, and V (cost of storage is $mp + np + p$) and that of A (cost of storage is mn, but only some fraction of this if A is sparse). While the construction of A_k is expensive (requiring in most cases the SVD of A), it can be approximated very accurately via *randomized algorithms*, which requires only the SVD of a smaller matrix. Various randomized algorithms for constructing the low-rank SVD and related factorizations are described in [13]. Techniques for computing the low-rank SVD of a matrix rely on a simple principal. An orthonormal matrix $Q \in \mathbb{R}^{m \times r}$ (with $r = k + l$ where l is a small oversampling parameter, e.g., $l = 5$), is computed such that $QQ^T A \approx A$. If in fact r is large enough so that $QQ^T A = A$, the range of A is a subset of the range of Q. Thus, when $QQ^T A \approx A$, we expect the range of Q to capture a good portion of the range of A, a statement which can be made rigorous with some analysis. In this case, we form the smaller matrix $B = Q^T A$, where $B \in \mathbb{R}^{r \times n}$, possibly much smaller than the $m \times n$ matrix A. Instead of performing the SVD on A, we can obtain the SVD of $B = U \Sigma V^T$. If $A \approx QQ^T A = QB$, then $A \approx (QU) \Sigma V^T$ and the later will form a low-rank SVD approximation for A (if we only take the first k singular vectors and values of the corresponding factorization). Notice that when A is rectangular, the eigendecomposition of the BB^{T} or $B^{\mathrm{T}}\ B$ matrices can be used to construct the approximate low-rank approximation of A.

A separate problem is the construction of a suitable matrix Q from A. Again, the idea is to construct as small (in terms of column number) as possible Q with orthonormal columns, such that Q captures a good chunk of the range of A. When A is a matrix of known rank k then (in MATLAB notation), simply setting $Q = qr(A\Omega, 0)$ where $\Omega \in \mathbb{R}^{n\times(k+l)}$ is a GIID matrix, with l a small over-sampling parameter, produces a valid matrix Q for projection. When the tail singular values (those smaller than σ_k) are still expected to be significant, a power sampling scheme turns out to be effective. Instead of setting $Y = A\Omega$ and performing QR of Y, we use the matrix $Y = \left(AA^T\right)^q A\Omega$ with q ≥ 1. Plugging in the SVD of A, we obtain $\left(AA^T\right)^q A = U\Sigma^{2q+1}V^T$, which has the same eigenvectors as A but much faster decaying singular values. Care must be taken when taking powers of matrices, to prevent multiplying matrices whose singular values are greater than one in magnitude. However, when the rank of the matrix A is not known, it is hard to use this approach, since the optimal size of the random matrix Ω to use would not be known. In this situation, a blocked algorithm can be employed [14], where on output with user supplied $\epsilon > 0$ parameter, an orthonormal matrix Q and matrix B are produced such that $||QB - A|| < \epsilon$ where $B = Q^T A$. Then, any number of standard low-rank matrix factorizations can be computed by operating on the matrix B instead of A. The basic steps of the proposed algorithm are given in **Figure 6**. We note that the resulting Q matrix can be utilized also for purposes of model reduction (e.g., one can use the reduced linear system $Q^T Ax = Q^T b$ as an approximation to the full system $Ax = b$). That is, one can use the reduced linear system $Q^T Ax = Q^T b$ as an approximation to the full system $Ax = b$ (or to replace A and b with the projected values in the least squares formulation); which has applications to e.g. accelerate image deblurring. The construction of Q for large matrices can in practice be done in parallel by employing the algorithm in [13] over row blocks of the matrix, as illustrated in **Figure 6**.



```
function $[Q, B] = \mathtt{randQB\_pb}(A, \varepsilon, q, b)$
for $i = 1, 2, 3, \ldots$
    $\Omega_i = \mathtt{randn}(n, b)$.
    $Q_i = \mathtt{orth}(A\Omega_i)$.
    for $j = 1 : q$
        $Q_i = \mathtt{orth}(A^T Q_i)$.
        $Q_i = \mathtt{orth}(AQ_i)$.
    end for
    $Q_i = \mathtt{orth}(Q_i - \sum_{j=1}^{i-1} Q_j Q_j^T Q_i)$
    $B_i = Q_i^T A$
    $A = A - Q_i B_i$
    if $\|A\| < \varepsilon$ then stop
end while
Set $Q = [Q_1 \cdots Q_i]$ and $B = [B_1^T \cdots B_i^T]^T$.
```

Figure 6. A *blocked* and *adaptive* version of the accuracy enhanced QB algorithm proposed in [14].

The rank-k SVD $\left(A_k = U_k \sum_k V_k^T\right)$ of a general $m \times n$ matrix A yields an optimal approximation of rank k to A, both in the operator (spectral) and Frobenius norms. On the other hand, even if A is a sparse matrix, the $m \times k$ and $n \times k$ factors U_k and V_k are typically dense. This means that if the matrix is approximately p percent filled, the matrix will have approximately $N = \left[\frac{p}{100} m \times n\right]$ nonzeros. On the other hand, the rank k SVD will consist of approximately $mk + k + nk$ nonzeros. For growing rank k, this quantity will quickly approach and even exceed N. Thus, even though the low-rank SVD is optimal for a given rank k, the choice of rank may be limited to relatively low values with respect to min(m, n) for sparse matrices, in order to achieve any useful compression ratios. (Of course, the usefulness of low-rank SVD representation is not simply limited to compression; indeed they are useful, e.g., for low-dimensional data projections; but the utility of a low-rank approximation is greatly reduced once the storage size of the factors exceeds that of the original matrix). Yet another aspect of the SVD which may be problematic is the difficulty in interpreting the eigenvectors present in U_k and V_k. While in many applications these have distinct meanings, they are not often easy to interpret for a particular data set.

It is thus plausible, in the above context, to look for factorizations which may not be optimal for rank k, but which may preserve useful properties of A such as sparsity and non-negativity, as well as allow easier interpretation of its components. Such properties may be found in the one- and two-sided interpolative decompositions and the CUR decomposition based on the pivoted QR decomposition. If we stop the QR procedure after the first k iterations, we obtain:

$$A(:,J_c) = \begin{array}{cc} & \begin{array}{cc} k & r-k \end{array} \\ m & \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \end{array} \times \begin{array}{cc} & n \\ k & \multirow{2}{*}{$\begin{bmatrix} S_1 \\ S_2 \end{bmatrix}$} \\ r-k & \end{array} = Q_1 S_1 + Q_2 S_2. \quad (16)$$

$$S_1 = \begin{array}{cc} & \begin{array}{cc} k & n-k \end{array} \\ k & \begin{bmatrix} S_{11} & S_{12} \end{bmatrix} \end{array} \text{ and } S_2 = \begin{array}{cc} & \begin{array}{cc} k & n-k \end{array} \\ k & \begin{bmatrix} 0 & S_{22} \end{bmatrix} \end{array} \quad (17)$$

$$\left(\text{i.e.,}\, S = \begin{array}{c|cc} & k & n-k \\ \hline k & S_{11} & S_{12} \\ r-k & 0 & S_{22} \end{array},\right) \quad (18)$$

$$A(:,J_C) = Q_1\begin{bmatrix} S_{11} & S_{12} \end{bmatrix} + Q_2\begin{bmatrix} 0 & S_{22} \end{bmatrix} = m\begin{bmatrix} \overset{k}{Q_1 S_{11}} & Q_1 S_{12} \overset{n-k}{+} Q_2 S_{22} \end{bmatrix}.$$

From this formulation, we set

$$C := A(:,J_C(1:k)) = Q_1 S_{11}.$$
$$Q_1 S_1 = \begin{bmatrix} Q_1 S_{11} & Q_1 S_{12} \end{bmatrix} = Q_1 S_{11}\begin{bmatrix} I_k & S_{11}^{-1} S_{12} \end{bmatrix} = C\begin{bmatrix} I_k & T_l \end{bmatrix},$$

where T_l is the solution to the matrix equation $S_{11}T_l = S_{12}$ (which if solved for T_l a column at a time, is simply a set of linear systems). It follows that we can write:

$$A \approx CV^T, \quad \text{where} \quad V^T = [I_k \quad T_l]P^T. \tag{19}$$

The one-sided ID of (rank k) is the approximate factorization:

$$\begin{array}{cccc} A & \approx & A(:,J_c(1:k)) & V^T, \\ m \times n & & m \times k & k \times n \end{array} \tag{20}$$

where we use a *partial column skeleton* $C = A(:,J_c(1:k))$ of a subset of the columns of A and V is a well-conditioned matrix. Clearly, C simply represents a subset of the columns of A chosen based on the pivoting strategy used in the QR factorization. The typical pivoting strategy is to choose the permutation matrix P (which simply dictates the re-arrangement of the columns of A) such that if S_{22} above is omitted, yielding:

$$AP \approx Q_1[S_{11} \quad S_{12}] + Q_2[0 \quad 0] = Q_1\tilde{S},$$

then the components of $\tilde{S}$ satisfy $|\tilde{s}_{11}| \geq |\tilde{s}_{22}| \geq \cdots \geq |\tilde{s}_{nm}|$. Several other pivoting strategies can be employed and each will yield a somewhat different re-arrangement of the columns of A.

Once the single-sided ID is defined, the two-sided ID can be constructed simply by obtaining a one-sided ID of A and that of A^T. A set of select columns of A^T obtained by this procedure, will be the same as the set of select rows of A. Thus, we can write the two-sided ID of (rank k) as:

$$\begin{array}{ccccc} A & \approx & W & A(J_r(1:k),J_c(1:k)) & V^T, \\ m \times n & & m \times k & k \times k & k \times n \end{array} \tag{21}$$

The procedure for the construction of the interpolative decompositions can be accelerated by means of randomization, just like for the low-rank SVD. This is possible by virtue of the result below [13].

Lemma 1 *Let $\tilde{\Omega} \in \mathbb{R}^{l \times m}$ be a matrix with GIID entries. Then, for any $a \in \mathbb{R}^m$, we have that $E\left[\frac{\|\tilde{\Omega}a\|^2}{\|a\|^2}\right] = l$ and $Var\left[\frac{\|\tilde{\Omega}a\|^2}{\|a\|^2}\right] = 2l$.*

Suppose, A is $m \times n$ and we draw a $l \times m$ GIID matrix $\tilde{\Omega}$. Suppose, we then form the $l \times n$ matrix $Z = \tilde{\Omega}A$. Then, $E\left[\frac{\|Z(:,j)\|^2}{\|A(:,j)\|^2}\right] = l$. As the pivoting result depends heavily on the ratio of the individual column norms of A with respect to one another, the above result tells us that the ratio of column norms is roughly preserved in a matrix resulting from the multiplication of the original matrix by a Gaussian random matrix from the left. As the product matrix consists of fewer rows than the original matrix, the pivoted QR factorization is correspondingly cheaper to perform on the product matrix Z than on A, while the resulting permutation matrix (really the re-arrangement vector) will be similar for both cases.

The two-sided ID allows us to construct the popular Column/Row skeleton CUR (rank k) decomposition:

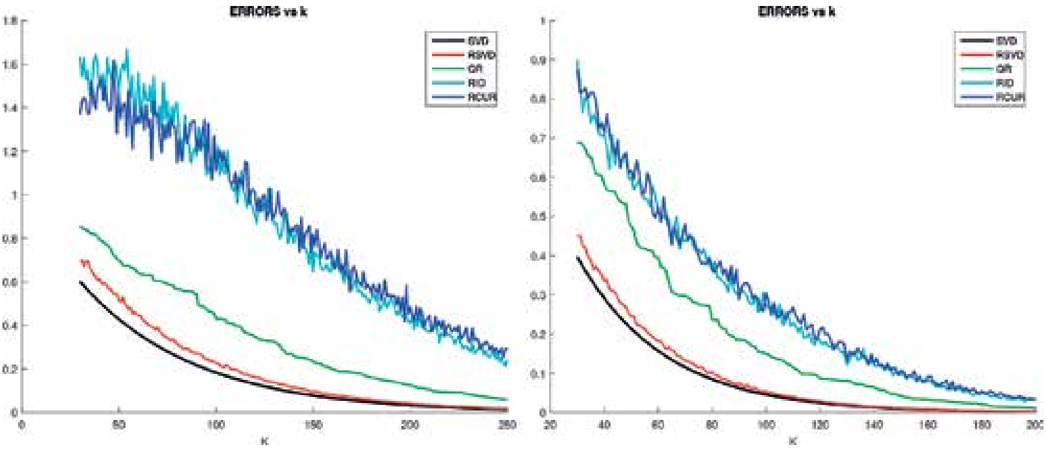


Figure 7. Relative errors for RSVD, ID, and CUR decompositions of rank k for matrices with two different rates of singular value decay (slower, faster).

$$\begin{array}{ccccc} A & \approx & C & U & R \\ m \times n & & m \times k & k \times k & k \times n \end{array} \tag{22}$$

Suppose, we compute a two-sided rank k ID factorization forming the $k \times k$ column/row skeleton $A(J_r(1{:}\ k), J_c(1{:}\ k))$. Set:

$$C = A(:, J_C(1:k)) \quad \text{and} \quad R = A(J_r(1:k), :)$$

We then set this to equal the factors C and R in CUR:

$$CUR = A(:, J_c(1:k))UA(J_r(1:k), :) \approx A(:, J_c(1:k))V^T \tag{23}$$

where we take U to satisfy the system $UR = V^T$: In **Figure 7**, we compare the relative errors obtained with different approximations at the same rank. For matrices with mild singular value decay, the low-rank SVD obtained via a randomized scheme (with oversampling) gives significantly closer to optimal performance (to true truncated SVD) than other decompositions.

8. An introduction to Backus-Gilbert inversion

As previously mentioned, damped least-squares (DLS) techniques are commonly exploited to solve linear, discrete inversion problems, such as those encountered in seismic tomography [15, 16]. To break the nonuniqueness of the least-squares solution, DLS inversion schemes often rely on *ad hoc* regularization strategies (e.g., model norm damping or smoothing), designed to subjectively favor the model simplicity. However, in regions of poor seismic data coverage, DLS methods may lead to locally *biased* model solutions—potentially causing model misinterpretations [17]. In other words, DLS models may represent "biased averages" over the true-model parameters. Most geotomographical studies suffer from uneven data coverages, and thus are concerned by these averaging *bias* effects. For example, teleseismic body-wave raypaths irregularly sample the Earth's interior, because earthquakes typically are concentrated along oceanic ridges or subduction zones and seismometers are located over continental areas or oceanic islands.

A fundamentally different approach is that of linear Backus-Gilbert inversion [18–20], which belongs to the class of optimally localized averages (OLA) methods. In the discrete version of the Backus-Gilbert theory, one aims at evaluating (weighted) averages of the true-model parameters. That is, the Backus-Gilbert method seeks to determine unbiased model estimates. Over the past half century, many authors have considered that, in addition to being computationally very intensive, it could be a clumsy affair in the presence of data errors to practically implement the Backus-Gilbert method to (large-scale) tomographic applications [15, 21–23]. In the following, we aim to describe a recently developed—and (finally!) computationally tractable—*tomographic* approach [10] based on the Backus-Gilbert philosophy.

The SOLA (subtractive optimally localized averages) method [24, 25] is a variant of the original Backus-Gilbert approach, which has been exploited to solve helioseismic inversion problems [26, 27]. As a remark, Pijpers and Thompson [24] termed this alternative the SOLA method, though it may have been rediscovered independently by different authors [28, 29]. SOLA retains all the advantages of the original Backus-Gilbert method, but is much more computationally efficient and versatile in the construction of resolving (averaging) kernels. Recently, SOLA has been introduced and adapted to large-scale, linear and discrete "tomographic" problems by Zaroli [10]. We now briefly review the SOLA inversion scheme, tailored to seismic tomography.

In this section, let us slightly change the notations about linear inverse problems, to keep closer with those preferred in the geosciences community [10, 17]. Let us consider linear, discrete forward problems of the form:

$$\mathbf{d} = \mathbf{G}\mathbf{m} + \mathbf{n}, \tag{24}$$

where $\mathbf{d} = (d_i)_{1\leq i\leq N}$ denotes the data, $\mathbf{G} = \left(G_{ij}\right)_{1\leq i,j\leq N,M}$ the sensitivity matrix, $\mathbf{m} = \left(m_j\right)_{1\leq j\leq M}$ the true-model parameters, and $\mathbf{n} = (n_i)_{1\leq i\leq N}$ the noise. The sensitivity matrix elements are the partial derivatives of the data with respect to the model parameters: $G_{ij} = \partial d_i/\partial m_j$. Typically, in "large-scale" tomographic studies, one may have to deal with $M \gtrsim 10^5$ model parameters and $N \gtrsim 10^6$ data. Let us consider, without loss of generality, that the data are time-residuals, the model parameters are velocity anomalies, the model space is parametrized using regular-size cells (local and "orthonormal" parameterization), the noise is randomly drawn from a normal distribution $\mathcal{N}(0, \sigma_n)$, and the data covariance matrix is $\mathbf{C}_d = \sigma_n^2 \mathbf{I}_N$. For local and "irregular" parametrizations, the reader is referred to [10]. It is a common practice to normalize both the data and sensitivity matrix by the data errors; thus $\mathbf{C}_d = \mathbf{I}_N$.

One aims to find a model estimate, $\widehat{\mathbf{m}}$, that can be expressed as a linear combination of the data:

$$\widehat{\mathbf{m}} = \widehat{\mathrm{G}}^{\dagger}\mathrm{d},$$

where the matrix $\widehat{\mathbf{G}}^{\dagger}$ denotes *some* generalized inverse. The model estimate can be decomposed as

$$\underbrace{\widehat{\mathbf{m}}}_{\text{model estimate}} = \underbrace{\widehat{\mathbf{R}}\mathbf{m}}_{\text{filtered true model}} + \underbrace{\widehat{\mathbf{G}}^{\dagger}\mathbf{n}}_{\text{propagated noise}}, \tag{25}$$

where

$$\widehat{\mathbf{R}}{=}\widehat{\mathbf{G}}^{\dagger}\mathbf{G}, \tag{26}$$

is often referred to as the model resolution matrix. The first term in right member of (25), $\widehat{\mathbf{R}}\mathbf{m}$, represents the filtered true model, and shows our inability, if $\widehat{\mathbf{R}} \neq \mathbf{I}_M$, to perfectly recover the true model. Here, we refer to the k-th row of the resolution matrix, $\widehat{\mathbf{R}}_{k.} = \left(\widehat{R}_{kj}\right)_{1\le j\le M}$, as the resolving kernel that linearly relates the k-th parameter estimate, $\widehat{m}_k$, to the true-model parameters:

$$\widehat{m}_k = \sum_{j=1}^{M} \widehat{R}_{kj} m_j, \quad \left(\text{ignoring the term of propagated noise}\right). \tag{27}$$

Therefore, we wish that $\widehat{\mathbf{R}}\mathbf{m}$ represents an *unbiased averaging* over the true model parameters, $\mathbf{m}$. This means that, for any parameter index $k \in [1, \ldots, M]$, we wish that $\widehat{\mathbf{R}}_{k}$. is non-negative and satisfies to

$$\sum_{j=1}^{M} \widehat{R}_{kj} = 1. \tag{28}$$

The second term in right member of (25), $\widehat{\mathbf{G}}^{\dagger}\mathbf{n}$, denotes the propagated noise (i.e. the propagation of data errors) into the model estimate. Robust model interpretations require accurate appraisals of model estimates, that is to compute and carefully analyze both $\widehat{\mathbf{R}}$ and the model covariance matrix

$$\mathbf{C}_{\widehat{m}} = \widehat{\mathbf{G}}^{\dagger}\mathbf{C}_d\left(\widehat{\mathbf{G}}^{\dagger}\right)^T. \tag{29}$$

As a remark, for DLS models this would also mean to quantify averaging bias effects (if any)—see [17]. The model estimate $\widehat{\mathbf{m}}$, resolution $\widehat{\mathbf{R}}$, and covariance $\mathbf{C}_{\widehat{m}}$ can be inferred from the generalized inverse $\widehat{\mathbf{G}}^{\dagger}$; efficiently computing the full generalized inverse is then crucial for any linear inverse problem. As we shall see, in the "SOLA Backus-Gilbert" approach the generalized inverse is directly determined.

The original Backus-Gilbert scheme consists in constructing the most peak-shaped resolving kernel (peaked around each model parameter location), while moderating at most the propagated noise into the model estimate. The key idea in the SOLA method is to specify an a *priori* "target form" for each resolving (averaging) kernel. One needs to specify M target resolving-kernels (hereafter, target kernels) such that their spatial extent represents some *a priori* estimate of the spatial resolving-length (around each parameter location). As an example, for 2-D tomographic studies the simplest target form could be circular (isotropic resolving-length); each target kernel would be constant inside such a circle and zero outside. Rather than minimizing the spread of each resolving kernel, as in the original Backus-Gilbert formulation, in the SOLA approach one aims at minimizing the integrated squared difference between each resolving kernel and its associated target kernel. Each *row* of the SOLA generalized inverse is individually computed by solving a specific minimization problem—the full computation of $\widehat{\mathbf{G}}^{\dagger}$ is then extremely parallel. The k-th row, $\widehat{\mathbf{G}}^{\dagger}_{k.} = \left(\widehat{G}^{\dagger}_{ki}\right)_{1 \le i \le N}$, is found such that:

$$\min_{\widehat{\mathbf{G}}^{\dagger}_{k.}} \quad \underbrace{\textstyle\sum_{j=1}^{M} \left(\widehat{R}_{kj} - T_j^{(k)}\right)^2}_{\text{resolution misfit}} + \eta_k^2 \quad \underbrace{\sigma^2_{\widehat{m}_k}}_{\text{model variance}}, \quad \text{s.t.} \quad \sum_{j=1}^{M} \widehat{R}_{kj} = 1, \tag{30}$$

where η_k and $\mathbf{t}^{(k)} = \left(T_j^{(k)}\right)_{1 \le j \le M}$ are the k-th tradeoff parameter (resolution misfit *versus* model variance) and target resolving-kernel vector, respectively; k is the index of considered model parameter. Because of the additional constraint in (30), the k-th parameter estimate, $\widehat{m}_k$, is expected to be *unbiased* (provided that its corresponding resolving kernel is (mostly) non-negative)—so for the model estimate $\widehat{\mathbf{m}}$. Though not strictly necessary, here all M target kernels are imposed to be unimodular:

$$\sum_{j=1}^{M} T_j^{(k)} = 1, \quad \forall k \in [1, \cdots, M]. \tag{31}$$

The system to be solved for the k-th row of the SOLA generalized inverse then writes as follows:

$$\left(\mathbf{G}\,\mathbf{G}^T + \eta_k^2 \mathbf{I}_N\right)\widehat{\mathbf{G}}^{\dagger}_{k.} = \mathbf{G}\mathbf{t}^{(k)}, \quad \text{s.t.} \quad \sum_{j=1}^{M} \sum_{i=1}^{N} \widehat{G}^{\dagger}_{ki} G_{ij} = 1. \tag{32}$$

As a remark, since only a single (k-th) parameter index is treated at a time in (32), it could be difficult to ensure that all M selected values for the tradeoff parameters ($\eta^{(k)}$) would lead to "globally coherent" model solutions. However, it seems [10, 17] that globally coherent tomographic images can be obtained when using: (1) target kernels whose size is tuned to the spatially irregular data coverage (for instance using seismic ray-paths density as a proxy for the spatial variations of the local resolving-length); and (2) constant-valued tradeoff parameters, that is:

$$\eta_k = \eta, \quad \forall k \in [1, \cdots, M]. \tag{33}$$

In practice, it seems that η may (roughly) be determined from analyzing a few curves of tradeoff between $\sum_j \left(\widehat{R}_{kj} - T_j^{(k)}\right)^2$ and $\sigma^2_{\widehat{m}_k}$, for some randomly chosen parameter index (k). Let us now define the following quantities [10, 17, 30]:

$$\begin{cases} \mathrm{x}^{(\mathrm{k})} & = & \left(x_i^{(k)}\right)_{1\le i\le N}, & x_i^{(k)} = \widehat{G}^{\dagger}_{ki} \\ \widehat{\mathrm{x}}^{(\mathrm{k})} & = & \left(x_i^{(k)}\right)_{2\le i\le N} & c_i = \sum_{j=1}^{M} G_{ij} \\ \mathrm{c} & = & (c_i)_{1\le i\le N}, & \\ \widehat{\mathrm{c}} & = & (c_i/c_1)_{2\le i\le N} & \\ \mathrm{e}_1 & = & (\delta_{i1})_{1\le i\le N} & \\ \mathbf{B} & = & \begin{pmatrix} -\widehat{\mathrm{c}}^{\mathrm{T}} \\ \mathbf{I}_{N-1} \end{pmatrix} & \\ \mathbf{Q}^{(\eta)} & = & \begin{pmatrix} \mathbf{G}^{\mathrm{T}}\mathbf{B} \\ -\eta\widehat{\mathrm{c}}^{\mathrm{T}} \end{pmatrix} & \\ \mathbf{y}^{(k,\eta)} & = & \begin{pmatrix} \mathrm{t}^{(k)} - c_1^{-1}\mathbf{G}^T\mathrm{e}_1 \\ -c_1^{-1}\eta \end{pmatrix}, & \end{cases} \tag{34}$$

where c_1 is assumed to be nonzero and δ denotes the Kronecker symbol. Solving (32) therefore consists in solving for $\widehat{\mathbf{x}}^{(k)}$ the following normal equations:

$$\begin{pmatrix} \mathbf{Q}^{(\eta)} \\ \eta\mathbf{I}_{N-1} \end{pmatrix} \widehat{\mathrm{X}}^{(k)} = \begin{pmatrix} \mathbf{y}^{(k,\eta)} \\ \mathbf{0}_{N-1} \end{pmatrix}, \tag{35}$$

using for instance the LSQR algorithm [31], and then to infer the final solution $\mathrm{x}^{(\mathrm{k})}$ (i.e., the k-th row of the SOLA generalized inverse) from $\widehat{\mathbf{x}}^{(k)}$ such that:

$$\mathrm{x}^{(k)} = \mathbf{B}\widehat{\mathrm{x}}^{(k)} + c_1^{-1}\mathrm{e}_1. \tag{36}$$

Last, but not least, we now aim to discuss about the computational efficiency of the SOLA approach for computing the full generalized inverse (see [10]). First, the rows of the generalized inverse matrix can be computed in *parallel* on P processors, so that computing all M rows would take $t \times M/P$ CPU-time, where t is the average CPU-time to numerically solve (35). A crucial point is that the matrix $\mathbf{Q}^{(\eta)}$, of size $(M+1) \times (N-1)$, does *not* depend on the parameter index (k), so that it does not need to be recomputed M times—as it was required in the original Backus-Gilbert approach (see [10]). The vector $\mathbf{y}^{(k,\eta)}$ has to be recomputed M times, but that task is computationally cheap. $\mathbf{Q}^{(\eta)}$ and $\mathbf{y}^{(k,\eta)}$ can easily be reconstructed if one aims at investigating different η values (only the last row of $\mathbf{Q}^{(\eta)}$ and last element of $\mathbf{y}^{(k,\eta)}$ depend on

η). Finally, simply re-ordering the rows of the sensitivity matrix **G** (and corresponding data), such that the *first* row of **G** is the sparsest one, allows the matrix $\mathbf{Q}^{(\eta)}$ to be almost as *sparse* as G —this sparsity property is very useful when solving (35), in terms of storage, efficiency of the LSQR algorithm, and memory footprint.

Figure 8 shows an example of the SOLA method applied to global-scale seismic tomography [10], for which there are M = 38, 125 model parameters and N = 79, 765 data (teleseismic shear-wave time-residuals). Tomographic images represent isotropic, 3–D shear-wave velocity perturbations within the whole Earth's mantle (with respect to some reference, radial absolute velocity model). **Figure 8a** and **b** displays the tomographic model $\widehat{\mathbf{m}}$, at about 600 km depth, and its uncertainty $\sigma\widehat{m}$ computed as

$$\sigma_{\widehat{m}} = \left(\sigma_{\widehat{m}_k}\right)_{1\le k\le M'} \quad \sigma_{\widehat{m}_k} = \sqrt{\sum_{i=1}^{N}\left(\widehat{G}_{ki}^{\dagger}\right)^2}, \tag{37}$$

(since the data are normalized by their errors), respectively. The form of each target kernel is that of a 3-D spheroid, corresponding to *a priori* lateral and radial resolving lengths that may

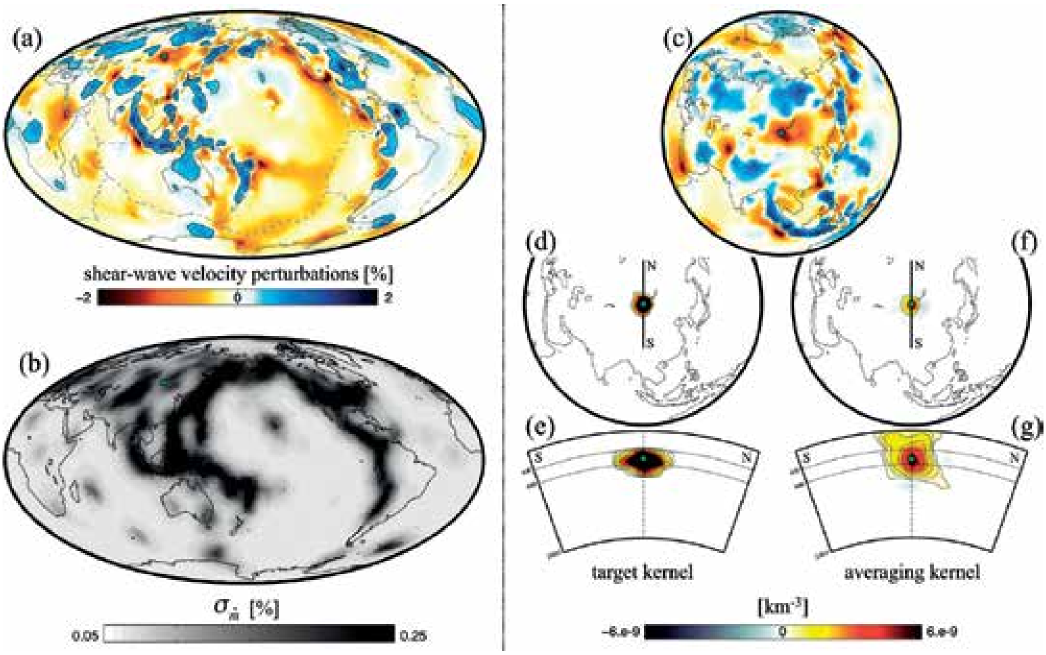


Figure 8. Example of a global geotomographical model and its associated resolution and uncertainty, obtained from using a "SOLA Backus-Gilbert" inversion approach [10]. (a) Model estimate, $\widehat{\mathbf{m}}$, shown at 600 km depth; (c) model uncertainty, $\sigma_{\widehat{m}}$, shown at 600 km depth; (b) zoom-in on $\widehat{\mathbf{m}}$ (600 km depth) around the k'-th parameter location, i.e., the green dot; (d, e) and (f, g) horizontal (600 km depth) and vertical cross-sections through the k'-th target (spheroid shape) and averaging kernels, respectively.

be expected locally, at best, given the data coverage. Let us focus on the k'-th model parameter, marked by a green dot in **Figure 8**; a zoom-in on the tomographic model is shown in **Figure 8c**. Horizontal (600 km depth) and vertical cross-sections through the k'-th target kernel are displayed in **Figure 8d** and **e**, respectively. The corresponding k'-th resolving (averaging) kernel is similarly displayed in **Figure 8f** and **g**.

Finally, the "SOLA Backus-Gilbert" approach, introduced and adapted to large-scale, linear, discrete tomographic problems by Zaroli [10], allows to efficiently compute unbiased models, including their full resolution and covariance—enabling quantitative model interpretations [17].

9. Conclusion

In this work, we have presented several techniques useful to the practitioner in the field of inverse problems, with the aim to give an idea of when and how these techniques should be employed for various linear and nonlinear applications. We have discussed techniques such as sparse matrix storage, the use of pivoted factorizations for direct solves, ℓ_2, ℓ_1 and intermediate penalty-based regularization strategies, nonlinear least squares problems, the construction and use of low-rank factorizations, and an application of the Backus-Gilbert inversion approach tailored to seismic tomography.

Author details

Sergey Voronin[1]* and Christophe Zaroli[2]

*Address all correspondence to: svoronin@i-a-i.com

1 Intelligent Automation Inc, Rockville, MD, USA

2 Institut de Physique du Globe de Strasbourg, Université de Strasbourg, EOST/CNRS, France

References

[1] Golub GH, Van Loan CF. Matrix Computations. Vol. 3. JHU Press; 2012

[2] Bell T, McKenzie B. Compression of sparse matrices by arithmetic coding. In: Data Compression Conference, 1998 (DCC'98) Proceedings; IEEE; 1998. pp. 23-32

[3] Candès EJ, Wakin MB. An introduction to compressive sampling. IEEE Signal Processing Magazine. 2008;**25**(2):21-30

[4] Voronin S, Zaroli C, Cuntoor NP. Conjugate gradient based acceleration for inverse problems. GEM-International Journal on Geomathematics. 2017;**8**(2):219-239

[5] Beck A, Teboulle M. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. SIAM Journal on Imaging Sciences. 2009;**2**(1):183-202

[6] Voronin S, Woerdeman HJ. A new iterative firm-thresholding algorithm for inverse problems with sparsity constraints. Applied and Computational Harmonic Analysis. 2013;**35**(1):151-164

[7] Voronin S, Chartrand R. A new generalized thresholding algorithm for inverse problems with sparsity constraints. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2013; IEEE; 2013. pp. 1636-1640

[8] Voronin S, Daubechies I. An iteratively reweighted least squares algorithm for sparse regularization. arXiv preprint: arXiv:1511.08970. 2015

[9] Wang Y, Yang J, Yin W, Zhang Y. A new alternating minimization algorithm for total variation image reconstruction. SIAM Journal on Imaging Sciences. 2008;**1**(3):248-272

[10] Zaroli C. Global seismic tomography using Backus-Gilbert inversion. Geophysical Journal International. 2016;**207**(2):876-888

[11] Hansen PC. The L-Curve and its Use in the Numerical Treatment of Inverse Problems. IMM, Department of Mathematical Modelling, Technical University of Denmark; 1999

[12] Eckart C, Young G. A principal axis transformation for non-Hermitian matrices. Bulletin of the American Mathematical Society. 1939;**45**(2):118-121

[13] Voronin S, Martinsson P-G. Rsvdpack: Subroutines for computing partial singular value decompositions via randomized sampling on single core, multi core, and gpu architectures. arXiv preprint: arXiv:1502.05366, 2:16. 2015

[14] Martinsson P-G, Voronin S. A randomized blocked algorithm for efficiently computing rank-revealing factorizations of matrices. SIAM Journal on Scientific Computing. 2016; **38**(5):S485-S507

[15] Aster RC, Borchers B, Thurber C. Parameter Estimation and Inverse Problems. Revised ed. Elsevier; 2012

[16] Nolet G. A Breviary of Seismic Tomography. Cambridge, UK: Cambridge University Press; 2008

[17] Zaroli C, Koelemeijer P, Lambotte S. Toward seeing the earth's interior through unbiased tomographic lenses. Geophysical Research Letters. 2017;**44**(22):11399-11408. DOI: 10.1002/2017GL074996

[18] Backus G, Gilbert JF. Numerical applications of a formalism for geophysical inverse problems. Geophysical Journal of the Royal Astronomical Society. 1967;**13**:247-276

[19] Backus G, Gilbert JF. The resolving power of gross earth data. Geophysical Journal of the Royal Astronomical Society. 1968;**16**:169-205

[20] Backus G, Gilbert JF. Uniqueness in the inversion of inaccurate gross earth data. Philosophical Transactions of the Royal Society A. 1970;**266**(1173)

[21] Menke W. Geophysical Data Analysis: Discrete Inverse Theory. Revised ed. San Diego: Academic Press; 1989

[22] Parker RL. Geophysical Inverse Theory. Princeton: Princeton University Press; 1994

[23] Trampert J. Global seismic tomography: The inverse problem and beyond. Inverse Problems. 1998;**14**:371-385

[24] Pijpers FP, Thompson MJ. Faster formulations of the optimally localized averages method for helioseismic inversions. Astronomy and Astrophysics. 1992;**262**:L33-L36

[25] Pijpers FP, Thompson MJ. The sola method for helioseismic inversion. Astronomy and Astrophysics. 1994;**281**:231-240

[26] Jackiewicz J, Birch AC, Gizon L, Hanasoge SM, Hohage T, Ruffio J-B, Svanda M. Multichannel three-dimensional SOLA inversion for local helioseismology. Solar Physics. 2012;**276**:19-33

[27] Rabello-Soares MC, Basu S, Christensen-Dalsgaard J. On the choice of parameters in solar-structure inversion. Monthly Notices of the Royal Astronomical Society. 1999;**309**: 35-47

[28] Larsen RM, Hansen PC. Efficient implementations of the sola mollifier method. Astronomy and Astrophysics Supplement Series. 1997;**121**:587-598

[29] Louis AK, Maass P. A mollifier method for linear operator equations of the first kind. Inverse Problems. 1990;**6**:427-490

[30] Nolet G. Solving or resolving inadequate and noisy tomographic systems. Journal of Computational Physics. 1985;**61**:463-482

[31] Paige CC, Saunders MA. Lsqr: An algorithm for sparse, linear equations and sparse least squares. ACM Transactions on Mathematical Software. 1982;**8**:43-71

Chapter 4

Evaluating the Hypersonic Leading-Edge Phenomena at High Reynolds and Mach Numbers

Frederick Ferguson, Julio Mendez and
David Dodoo-Amoo

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.72320

Abstract

Computational Fluid Dynamics (CFD) solutions have played an important role in the design and evaluation of complex problems where analytical solutions are not available. Among many practical applications, hypersonic flows have been an area of intense research because of the important challenges found in this flow regime. The numerical study conducted herein, focuses on solving the hypersonic flat plate problem under realistic conditions, at high Reynolds and Mach numbers. The numerical scheme implemented in this study solves the two-dimensional unsteady Navier Stokes Equations, using a novel technique called Integro-Differential Scheme (IDS) that combines the traditional finite volume and the finite difference methods. Moreover, this scheme is built on the premise of reducing the numerical errors through the implementation of a consistent averaging procedure. Unlike other numerical approaches, where free molecular effects are considered, this study enforces no-slip and fixed temperature as boundary conditions. The IDS approach accurately predicted the physics in the vicinity of the hypersonic leading edge at such realistic conditions. Even though there are slight discrepancies between the numerical solution and the available experimental data, the IDS solution revealed some interesting details about the flow field that was previously undiscovered.

Keywords: hypersonic flows, computational fluid dynamics, flat plate, viscous-inviscid interactions

1. Introduction

The flow over a flat plate is a classic yet fundamental fluid dynamic problem. Although the flow boundaries appear to be simple, the resulting flow field depends greatly on the prescribed free stream conditions. Of course, the free stream conditions are mainly defined by the Mach and Reynolds numbers as well as the ratio of specific heats. It is the ranges at which the free stream conditions are set that dictate the physics of the resulting flow field over the flat plate, and the complexities associated with it. Herein lie the many technical challenges of predicting the flat plate flow field. For example, at low Reynolds number and for subsonic Mach numbers at constant specific heats ratio of 1.4, the resulting flat plate only encourages the growth of a laminar boundary layer. Simulating such flow fields is relatively simple. As the Reynolds number increases, the boundary layer transitions to turbulent and the flow field becomes more challenging to simulate numerically. In the cases where the Reynolds number gets in the order of several million, the Mach number gets into the Hypersonic range, and the ratio of specific heat gets closer to 1.2, making the flow field interactions get complicated, and numerical simulations become unpredictable.

This chapter is concerned with the flow field over a flat plate at hypersonic conditions and at high Reynolds number. Understanding the flow field dynamics at these conditions will provide aerospace designers valuable insights into the complex interactions found in space vehicles such as rockets, space shuttles as well as military applications, for instance, hypersonic and long-range missiles. Under these conditions, the major aerodynamic concerns are aerodynamic heating and shockwave boundary layer interactions. In addition, the flow field may consist of two flow regimes; one mainly governed by the kinetic flow theory and another governed by the continuum flow theory [1]. In the case of the flat plate, especially near the tip, it is speculated that the displacement thickness increases rather drastically, causing the flow to move upward, initiating a compression shock wave and the formation of a strong interaction region. A weak interaction region follows this region. The resulting flow field becomes even more complicated because of the complex dynamics associated with the two regions. This shock is called a *bow shock*, due to its characteristic curvature. The region between the surface and the shock wave is called the *shock layer* [2], refer to **Figure 1**. Further, the shock layer is divided into two sublayers, each dominated by either inviscid or viscous effect. The sub-layer closest to the plate surface is known as the *boundary layer*, and the outer sublayer is the so-called *entropy layer*. Typically, the boundary layer undergoes an important transition; usually from a laminar to a turbulent boundary layer.

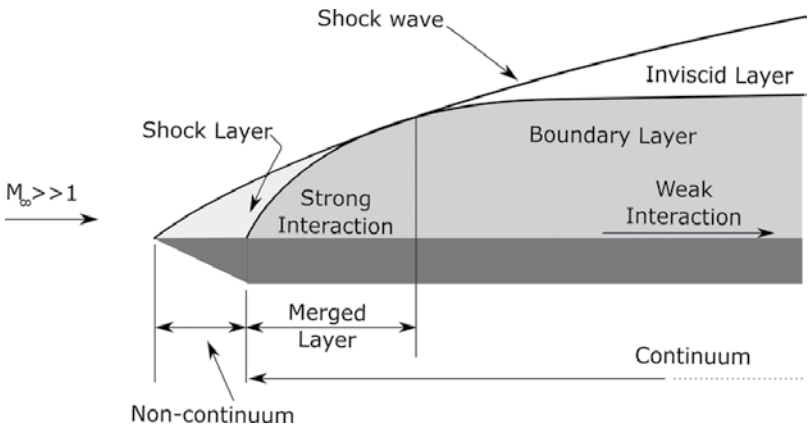


Figure 1. Sketch of the flat plate problem.

Two regions can also characterize the flow along the plate; one near the leading edge and another further away. In the leading-edge region, the viscous-inviscid interactions are very strong, and they affect both sublayers: the inviscid entropy and the viscous boundary sublayers. Further, this strong interaction results in the merging of the *entropy* and *boundary layer*. In contrast, further away from the leading edge, the inviscid-viscous interaction is weak, and the two sub-layers remain separated. The two zones that are mainly characterized by the inviscid-viscous interactions are referred to as the *strong* and *weak* interaction regions, respectively. The flow phenomena in the *strong* and *weak* interaction regions at the leading edge of the hypersonic plate problem are of paramount importance to this analysis. Because of the inherent complexity of the flow physics, analytical models are scarce, and reliable analyses can only be obtained exclusively by experiments and numerical simulations.

CFD emerged as a valuable tool for these types of flow studies. Nevertheless, the CFD tool must be capable of resolving sharp gradients while negotiating systems of partial differential equations of varying types. In other words, not only are the grids expected to be extremely fine to fully resolve the sharp gradients manifesting in these regions, the CFD schemes are also expected to remain computationally stable, accurate and timely.

Many numerical solutions have been proposed. Blottner [3] solved the boundary layer problem with finite chemical reactions using finite differences. In this study, 11 chemical species and 20 reactions were considered. Another numerical study was carried out by [4], where the full time dependent Navier Stokes Equations (NSE) were solved using particle-in-cell and fluid-in-cell computing methods. In addition, the study revealed that pressure gradient is appreciable near the leading edge. Unlike reference [3], the boundary conditions used in [4] were velocity slip and temperature jump at the surface plate. These types of boundary conditions are widely applied in rarefied hypersonic flows, which have been an active area of research. These types of flows are found near the leading-edge and experimental results suggest that strong interaction theory overpredicts the surface pressure [4]. These discrepancies are attributed to the transition between continuum and free molecular flow. An extensive comparison was presented by [5], where Direct Simulation Monte Carlo (DSMC) results were compared to the NSE solution in order to evaluate the accuracy of the NSE in this regime. They concluded that including the slip conditions improved the predicted values on the surface properties. However, knowledge about the Knudsen layer is required to properly define the slip conditions at the surface. Although the Knudsen number is small in the freestream, its value is considerably high close the surface where density gradients are large [5]. Under these scenarios, the continuum hypothesis of the NSE falls apart and the accuracy of the technique is no longer ensured. Furthermore, the numerical implementation of such boundary conditions requires further simplifications and assumptions. For example, the tangential and energy accommodation factors affect the CFD solutions. Tangential accommodation values of 0.5 seem to provide accurate results near the leading edge, whereas values between 0.75 and 1.0 yield the best agreement further along the plate [6]. The same author in another publication [7] claimed that the difficulty of defining these slip conditions is in determining the correct values for the coefficients mentioned above and other empirical terms required for the implementation.

The major objective of this book chapter is to numerically solve the hypersonic flow over a flat plate problem with a novel numerical method called *Integro-Differential Scheme* (IDS) [8]. This study ignores the slip and jump boundary conditions introduced by [7] and directly prescribes the boundary conditions applicable to the continuum flow theory. Based on the literature review presented above, the authors of this chapter suggested that the slip and jump conditions are more appropriate for use with the Burnet equations. Further, many literature reviews suggested that the hypersonic leading-edge phenomena are best explained through the use of the transition regime, which intersects continuum and free molecular flow theories, where Burnet equations are appropriate. Although, technical evidence exists to support this hypothesis [5] this demonstration is not the focus of this book chapter.

2. The governing equation

Numerical solutions of fluid dynamic problems are governed by conservation laws. These laws can be expressed mathematically either in the differential or the integral form. In the case of compressible fluid flows, these coupled laws form a closed system of partial differential equations that is called the system of Navier-Stokes Equations (NSE). Herein, the conservation of mass, momentum and energy principles in the integral form are of interest to this study, and they are expressed as follows:

$$\frac{\partial}{\partial t}\oiiint_{V} \rho dv + \oiint_{S} \left((\rho u)_i ds_i\right) = 0 \tag{1}$$

$$\frac{\partial}{\partial t}\oiiint_{V} (\rho u)_k dv + \oiint_{S} \left((\rho u)_i ds_i\right) u_k = -\oiint_{S} \left(p ds_i\right) + \oiint_{S} \tau_{ik} ds_i \tag{2}$$

$$\frac{\partial}{\partial t}\oiiint_{V} (\rho e) dv + \oiint_{S} \left((\rho e u)_i ds_i\right) = -\oiint_{S} \left(p u_i ds_i\right) + \oiint_{S} \left((\tau_{ik} u_k) ds_i\right) + \oiint_{S} \left(q_i ds_i\right) \tag{3}$$

In Eqs. (1)–(3) the symbols: ρ, u, t represent the density, the velocity components of an elementary control fluid element, and time, respectively. In addition, the symbols e, p, τ_{ij} and q_i in Eqs. (1)–(3) represent the internal energy, pressure, the stress tensor and the heat flux associated with an elementary control volume, respectively. Internal energy, pressure, stress tensor and heat flux are defined by Eqs. (4)–(7)

$$e = C_v T + \frac{1}{2} u_k u_k \tag{4}$$

$$p = \rho R T \tag{5}$$

$$\tau_{ik} = \mu \left(\frac{\partial u_i}{\partial x_k} + \frac{\partial u_k}{\partial x_i} - \frac{2}{3} \delta_{ik} \frac{\partial u_j}{\partial x_k} \right) \tag{6}$$

$$q_k = -k \frac{\partial T}{\partial x_k} \tag{7}$$

In Eq. (5), R is the gas constant. The symbols μ and k represent the viscous and thermal properties of the fluid of interest. For air, the viscosity of the fluid is evaluated using Sutherland's law and the thermal conductivity expression,

$$\mathrm{k} = \left(4.76 \times 10^{-6}\right) \mathrm{T}^{3/2}/(\mathrm{T} + 112.0) \tag{8}$$

In the case of 3D aerothermodynamics, the NSE (1–8) represent a closed system of five equation relative to five unknowns. These unknowns are called Primitive Variables (PV), and are defined in the vector form as follows:

$$\mathrm{PV} \;\coloneqq\; [\rho \;\; \mathrm{u} \;\; \mathrm{v} \;\; \mathrm{w} \quad \mathrm{T}]^{T} \tag{9}$$

The goal of any numerical solution to the NSE is to determine the primitive variables at every grid point. However, obtaining a unique solution to the NSE (1–8) requires the prescription of initial and boundary conditions.

Of course, the full set of the NSE (1–8) does not readily lend itself to analytical solutions. It is only in recent decades, with the advent of modern computers that the non-existence of analytical solutions to the NSE ceased to be a limitation to our understanding of the physics underpinning flow fields. Modern computers also gave birth to the many modern numerical methods capable of solving the NSE. Among these methods are the Conservation-Element Solution Element (CESE) method, Direct Numerical Simulation (DNS), Large Eddy Simulation (LES), Discontinuous Galerkin Methods (DGM) and the Integro-Differential Scheme (IDS). These computational methods have all been applied to the task of solving the NSE, and have all provided varying degrees of success when it comes to elucidating the details associated with the various flow field physics of interest to the engineers at realistic Reynolds and Mach numbers. This report highlights the IDS procedure of solving the NSE.

3. The IDS fluid model

Consider the Integro-Differential Model (IDM) as it is applied to the computational solution to the NSE (1–8). In general, the IDS solution of a given fluid dynamic problem is built on an interconnecting set of *spatial* and *temporal* fluid cells. In the Cartesian system of coordinates, a typical fluid cell is nothing more than a carefully chosen elementary rectangular prism, defined by the dimension; *dx*, *dy* and *dz*. It is the application of a specified fluid cell in relationship to the NSE equations that determines whether it becomes a *spatial* or a *temporal* cell. Nevertheless, consider the fluid cell illustrated in **Figure 2** where its implementation in the NSE is irrelevant at this time.

3.1. IDS cell properties

The Cartesian cell defined in **Figure 2**, has the following properties:

1. The rectangular prism has 6 elementary surfaces and each surface is defined through the use of 3 directional normals. Further, each normal is defined in either a positive or a

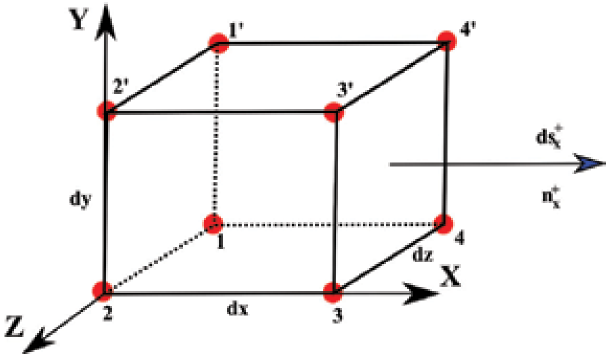


Figure 2. Spatial cell with notation at surface nodes.

negative direction in relationship to its respective axis. This definition is not unique to the IDM, but it is used here for illustrative purposes only

2. Each of the six elementary surfaces, *ds*, of the fluid cell, is considered a vector and is defined as follows: $\delta s_i^{\pm} = n_i^{\pm} d\xi_j d\eta_k$, where the indices *i*, *j*, and *k* vary from 1 to 3, representing the *x*, *y* or *z* coordinate direction. In addition, the area is considered a vector, having both direction and magnitude. Likewise, the volume of the elementary cell can be defined by $\delta v = \mathrm{dxdydz}$ for uniform grids. Note, this type of evaluation also works well for non-uniform and unstructured grids.

3. The fluid cell defined in **Figure 2** also allows for mass, momentum and energy fluxes to traverse its surfaces. At any given instance, the net spatial fluxes traversing a given surface are defined by a combination of their *inviscid* and *viscous* counterparts. The *inviscid* and *viscous* fluxes on the cell surfaces are defined by the symbols: E, E_{vis}, F, F_{vis}, G, and G_{vis}, representing the inviscid and viscous fluxes in the *x*, *y* and *z* directions, respectively.

4. In accordance with the IDM, the flux values are approximated from their edge quantities using their arithmetic averages, and are assumed to be located at the center of the respective surfaces. Consequently, all quantities evaluated on any of the cell surfaces are labeled as *averaged* quantities. The fluxes of interest on the cell surfaces are the average *inviscid* and *viscous* spatial fluxes on the $n_x^{\pm}$ surfaces, and they are defined by the symbols: $E_{Avg}^{Surf,\,nx\pm}$ and $E_{Vis,\;Avg}^{Surf,\,nx\pm}$. Likewise, the average *inviscid* and *viscous* fluxes on the $n_y^{\pm}$ and $n_z^{\pm}$ surfaces are defined by the symbols: $F_{Avg}^{Surf,\,nx\pm}$, $F_{Vis,\;Avg}^{Surf,\,nx\pm}$, $G_{Avg}^{Surf,\,nz\pm}$ and $G_{Vis,\;Avg}^{Surf,\,nz\pm}$, respectively. Having established the fact that the average cell flux quantities can only be defined on one of its elementary surfaces, the expressions needed for the evaluation of the spatial fluxes can now be summarily expressed as,

$$E = \delta s_i^{\pm} \left\{ \begin{array}{l} \rho u \\ \rho u^2 + p \\ \rho u v \\ \rho u w \\ (\rho e_T + p) u \end{array} \right\}_{Avg^S}, \quad E_{Vis} = \delta s_i^{\pm} \left\{ \begin{array}{l} 0 \\ \tau_{xx} \\ \tau_{xy} \\ \tau_{xz} \\ u\tau_{xx} + v\tau_{xy} + w\tau_{xz} - q_x \end{array} \right\}_{Avg^S} \tag{10}$$

where the subscripts in the right-hand terms represent the location and type of operations used in evaluating the required average quantities. Similarly, for the inviscid and viscous fluxes, F, F_{vis}, G, and $G_{vis.}$

5. In a similar manner, the average quantities within the cell volume, such as the time fluxes, U^{Cell}_{Avg}, and the rate of change of the time fluxes, $(\partial U/\partial t)^{Cell}_{Avg}$, are defined as

$$U^{Cell}_{Avg} = \left\{ \begin{matrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_T \end{matrix} \right\}_{Avg^v} , \left(\frac{\partial U}{\partial t}\right)^{Cell}_{Avg} = \frac{\partial}{\partial t} \left\{ \begin{matrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e_T \end{matrix} \right\}_{Avg^v} \tag{11}$$

3.2. Computing the cell properties

At this point, the concept of evaluating the volume and surface areas of an *elementary fluid cell* is fully formulated. However, the computation of the average *time* and *spatial* fluxes within and on the surface of an IDS cell is still not uniquely defined. For example, how are the *primitive variables*, which are defined at a point and the *average time fluxes* within the cell or the *spatial fluxes* on an elementary surface related? How is the flow field defined in relationship to the IDS fluid cell concept? To answer these questions and others, consider **Figure 2** once more. Assume the red dots in **Figure 2** represent the physical grid points in the flow field of interest, and at each of these points, the primitive variables are known. A typical elementary fluid cell is then built around eight such points, with each point separated by no more than one grid point. Using these assumptions, the average time fluxes within the cell can be computed from the arithmetic mean, as:

$$U^{Cell}_{Avg} \quad = \quad \frac{1}{8} \left[\left\{ \begin{matrix} \rho \\ u \\ v \\ w \\ T \end{matrix} \right\}^{1} + \left\{ \begin{matrix} \rho \\ u \\ v \\ w \\ T \end{matrix} \right\}^{1'} + \quad \dots \quad + \left\{ \begin{matrix} \rho \\ u \\ v \\ w \\ T \end{matrix} \right\}^{4'} \right] \tag{12}$$

Similarly, cell surface quantities, such as, the n_x^+ surface fluxes defined by $E^{Surf, nx+}_{Avg}$ can be computed as follows:

$$E^{Surf, nx+}_{Avg} = \frac{ds^+_{nx}}{4} \left[\left\{ \begin{matrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho e + p) \end{matrix} \right\}^{3}_{nx+} + \left\{ \begin{matrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho e + p) \end{matrix} \right\}^{3'}_{nx+} + \left\{ \begin{matrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho e + p) \end{matrix} \right\}^{4}_{nx+} + \left\{ \begin{matrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(\rho e + p) \end{matrix} \right\}^{4'}_{nx+} \right] \tag{13}$$

In a similar manner, the other inviscid flux quantities at the n_x^+, $n_y^\pm$, and $n_z^\pm$ surfaces can be found. Unfortunately, computing the viscous flux quantities; $E^{Surf, nx\pm}_{Vis, \ Avg}$, $F^{Surf, nx\pm}_{Vis, \ Avg}$, and $G^{Surf, nz\pm}_{Vis, \ Avg}$, are somewhat complicated and greater care is required. Refer to Ref. [8] for details. In summary, the IDM allows for the grid points, and the primitive variables allocated at those points to be used in uniquely formulating the elementary fluid cells and completely defining their flow field characteristics.

3.3. The IDS control volume and its properties

A typical IDS control volume in the 3D Cartesian system of coordinates is illustrated in **Figure 3**.

As can be observed, the IDS control volume consists of eight cells. The properties of the IDS control volume is as follows:

1. It is of interest to note that the centers of the eight cells form the vertex of an overlapping cell. This overlapping cell is called the *temporal* cell. As such, the control volume consists of eight neighboring cells that allow for the formation of a *temporal* cell. In other words, analogous to the manner in which the eight grid points formed the vortex of a given cell, so too, the center of eight neighboring cells formed the vortex of a *temporal* cell.

2. Also of interest to note is the fact that at the vertex of the *temporal* cell, the rate of change of the time fluxes are known. Consequently, at the center of the *temporal* cell, the average time rate of change of the time fluxes are computed as,

$$\left(\frac{\partial U}{\partial t}\right)_{\text{Avg}}^{\text{CV}} = \frac{1}{8}\left[\left(\frac{\partial U}{\partial t}\right)_{\text{Avg}}^{\text{Cell}_1} + \left(\frac{\partial U}{\partial t}\right)_{\text{Avg}}^{\text{Cell}_2} + \quad \cdots\cdots \quad + \left(\frac{\partial U}{\partial t}\right)_{\text{Avg}}^{\text{Cell}_8}\right] \tag{14}$$

3. Similarly, at the center of an IDS control volume, the average time fluxes are defined by the arithmetic averages

$$\mathrm{U}_{\text{Avg}}^{\text{CV}} = \frac{1}{8}\left[\mathrm{U}_{\text{Avg}}^{\text{Cell}_1} + \mathrm{U}_{\text{Avg}}^{\text{Cell}_2} + \quad \cdots\cdots \quad + \mathrm{U}_{\text{Avg}}^{\text{Cell}_8}\right] \tag{15}$$

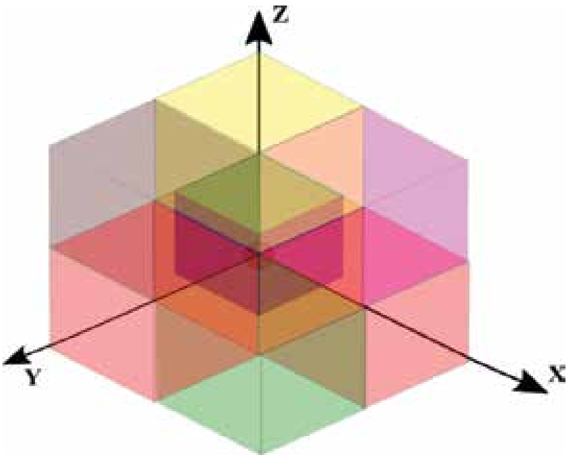


Figure 3. Illustration of control volume.

4. The IDS solution can only be advanced from within the *temporal* cells. Within a given *temporal*, the average temporal fluxes are updated as follows:

$$\mathrm{U}^{\mathrm{CV}}_{\mathrm{Updated,\,Avg}} = \mathrm{U}^{\mathrm{CV}}_{\mathrm{Avg}} + \left(\frac{\partial U}{\partial t}\right)^{\mathrm{CV}}_{\mathrm{Avg}} \delta t \tag{16}$$

where the symbol δt is the time increment, and methods for its computations are defined later in this report.

5. In the 3D Cartesian system of coordinates, the grids can be developed such that the center of the IDS control volume always overlaps with the center of a grid point. If this were the case, then the updated primitive variables can be computed from the expression

$$\begin{Bmatrix} \rho \\ u \\ v \\ w \\ T \end{Bmatrix}_{\mathrm{Updated}} = \begin{Bmatrix} \mathrm{U}_1 \\ \mathrm{U}_2/\mathrm{U}_1 \\ \mathrm{U}_3/\mathrm{U}_1 \\ \mathrm{U}_4/\mathrm{U}_1 \\ \left(\frac{\mathrm{U}_5}{\mathrm{U}_1} - \frac{1}{2}\left(\frac{\mathrm{U}_2^2}{\mathrm{U}_1^2} + \frac{\mathrm{U}_3^2}{\mathrm{U}_1^2} + \frac{\mathrm{U}_4^2}{\mathrm{U}_1^2}\right)\right)\gamma(\gamma - 1)\mathrm{M}_\infty^2 \end{Bmatrix}^{\mathrm{CV}}_{\mathrm{Updated,\,Avg}} \tag{17}$$

It is with this IDM in mind, where the spatial cell, the temporal cell and the control volume concepts are paramount, that the NSE are transformed into their *Integro-Differential* counterparts. In addition to this, Eqs. (14)–(16) highlight the main differences between the IDS and other CFD schemes, where the solution vector and the time rate are computed using local values, whereas the IDS computes the right-hand side terms from Eq. (16) through the use of Eqs. (14) and (15). From the computational perspective, the IDS performs more floating points operations per node and therefore, the method is computationally expensive. Roughly speaking, for 2D flows the method performs 8 times more floating-point operations. Nevertheless, the calculation of the viscous stresses and heat fluxes, Eq. (10), demands the evaluation of the stresses and heat fluxes at the faces of the control volumes, where the mean value theorem is used and hence, an extra averaged is required. Off course, for 3D fluid flows these operation increases because of the spanwise component where the averaging procedure is also implemented.

4. The Integro-differential scheme

Consider the IDM described in the preceding section for the special 3D Cartesian system of coordinates. If the NSE (1–3) were directly applied to a fluid element, such as the one illustrated in **Figure 2**, the analytical solution arising from this process, especially when the mean value principles are invoked, will yield the following transformational equations:

$$\left(\frac{\partial \rho}{\partial t}\right)_{\mathrm{Avg}} = -\left[\sum_{\mathrm{m}=1}^{6} (\rho \mathrm{u} \delta \mathrm{s})^{\pm}_{\mathrm{m}}\right] \Big/ \delta v \tag{18}$$

$$\left(\frac{\partial(\rho u_k)}{\partial t}\right)_{\text{Avg}} = -\left[\sum_{m=1}^{6}\left((\rho u_i + p\delta_{ik})\delta s\right)_m^{\pm} u_k - \sum_{m=1}^{6}(\tau_{ik}\delta s)_m^{\pm} u_k\right] \Big/ \delta v \tag{19}$$

$$\left(\frac{\partial(\rho e)}{\partial t}\right)_{\text{Avg}} = -\left[\sum_{m=1}^{6}\left((\rho e + p)u_i\delta s\right)_m^{\pm} - \sum_{m=1}^{6}(\tau_{ik}u_k\delta s)_m^{\pm} - \sum_{m=1}^{6}(q_i\delta s)_m^{\pm}\right] \Big/ \delta v \tag{20}$$

where the index *m*, *m* = 1, 6, defines the surfaces with positive and negative normal, respectively, and the indices *i* and *k*, go from 1 to 3, defining the coordinate directions. The technical challenges in computing Eqs. (14)–(16) lie in the careful and consistent manner in which the NSE auxiliary/closure Eqs. (4)–(8) are evaluated as they are applied to a fluid cell. It is worthwhile to repeat that Eqs. (4)–(8) must be evaluated in accordance with the requirements of each cell.

In the special case of 3D Cartesian systems with the spatial flow field domain defined on uniform cells, the viscous and inviscid fluxes can be expressed in vector form, as:

$$\left(\frac{\partial U}{\partial t}\right)_{Avg} + \frac{\Delta_{nx}^{\pm}(E - E_{vis})}{\Delta v} + \frac{\Delta_{ny}^{\pm}(F - F_{vis})}{\Delta v} + \frac{\Delta_{nz}^{\pm}(G - G_{vis})}{\Delta v} = 0 \tag{21}$$

where the U, E, E_{vis}, F, F_{vis}, G and G_{vis} vectors were defined earlier. The subscripts in Eq. (17) define the location and type of operations used in evaluating respective average quantities. In addition, the difference operators; $\Delta_{nx}^{\pm}$, $\Delta_{ny}^{\pm}$, and $\Delta_{nz}^{\pm}$ represent the difference in the surface fluxes across each cell, such that, the surface information of each cell are independently computed.

In summary, the integral form of the NSE equations (1–3) were analytically solved using the mean value theorem over an elementary control volume. The resulting solution was expressed in the form an *Integro-differential formulation* as described by Eq. (17). Finally, as in all explicit schemes, Eq. (16) can be used to compute the update solution vector U, such that, where the time fluxes and rate of change of the time fluxes vectors were defined in Eqs. (14) and (15). The symbol *δt* is the time increment and is computed by using the Courant-Friedrichs-Lewy (CFL) criterion. In this book chapter, the CFL criterion is computed with the aid of the expression [2],

$$\delta t = C\left[\frac{|u|}{\Delta x} + \frac{|v|}{\Delta y} + \frac{|w|}{\Delta y} + a\sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}} + \frac{2}{\mathrm{Re}_L}\nu\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}\right)\right]^{-1} \tag{22}$$

where *a* is the local speed of sound, *C* is the Courant number, and *γ* is the specific heat ratio, and ν is computed from the expression

$$\nu = \max\left[\left(\frac{4}{3}\mu, \quad (\gamma\mu/\mathrm{Pr})\right)\Big/\rho\right]$$

The typical values used for C in this analysis range as follows: $0.5 \leq C \leq 0.8$.

The IDS offers an important numerical advantage given that it is a very stable and accurate method. Further, the IDS provides a significant reduction in both *spatial* and *temporal* numerical dispersion through its use of the mean value theorem in computing the finite volume quantities. In the IDS approach, the time and spatial fluxes are appropriately approximate. In addition, the method has been shown to be consistent and has a minimum discretization error of order p = 2. A major advantage of the IDS method is that computations involving the compressible NSE are very stable, and no numerical oscillations are typically detected when the grid is fully refined. Another advantage of the IDS method is that it is suitable for solving both steady and unsteady flows at realistic Reynolds and Mach numbers. Experiences have shown that it is quite a satisfactory method for solving high Reynolds number flows, where the viscous regions are very thin, and shock boundary layer interactions are significant. For these flows, once the mesh is highly refined the IDS is able to resolve the flow physics within the viscous regions with a great order of accuracy.

5. The hypersonic flow over a flat plate

Consider the hypersonic flow over a 1 meter long flat plate at zero angle of attack. The freestream Mach number is set at 8.6, the Reynolds number set to 3.4757×10^6 (based on the plate length), the Prandtl number to 0.70 and the specific heat ratio, γ, to 1.4. These conditions are similar to those presented by [9] except that their experiments considered a slightly longer plate. Nevertheless, in comparison to this effort, the Reynolds number used in [9] was in perfect agreement, but the Mach number was slightly greater (approximately 0.93%). The boundary conditions were set as follows: free stream values were assigned to the inflow and the far field boundaries, the interior flow primitive variables were extrapolated to the exit plane, and three separate sets of conditions were assigned to the base of the domain. Symmetric boundary conditions were assigned to the leading and trailing edge gaps, and a combination of no-slip and fixed wall temperature assigned to the solid wall. It is of interest to mention here, in this effort the dimensionless temperature is set to 1.0 at the wall, whereas it was set to a value of 0.828 in the experimental study [9]

In efforts to obtain a grid independent solution, five sets of grids of sizes ranging from 1001 by 1001 nodes to 5001 by 16001 nodes in the streamwise and vertical direction, respectively, were studied. Since the gradients of the flow field parameters in the direction normal to the wall are sharper than those in the direction parallel to the wall, substantially finer grids were placed in the vertical direction [2]. In addition to the five sets of grids described above, an extra case, termed the modified grid, was also considered. This was done in efforts to more thoroughly evaluate the IDS capabilities in predicting the viscous dissipation effects inside the boundary layer. In the case of the modified grid, the height of the domain was reduced by half, resulting in an equivalent grid size of 6001 by 32001 nodes. This reduction in height effectively reduced the cell height by a factor of 2, resulting in a finer set of grids without a substantial increase in computational load.

The IDS flow field solutions resulting from the grid independence study are summarized in the **Figures 4** and **5**. Note, that in **Figures 4** and **5**, the modified grid is represented by the grid size of 6001 by 16001*. Note, **Figure 4** illustrates the behavior of the streamwise velocity component in the y-direction at a location of x = 0.5 m from the leading edge. A careful

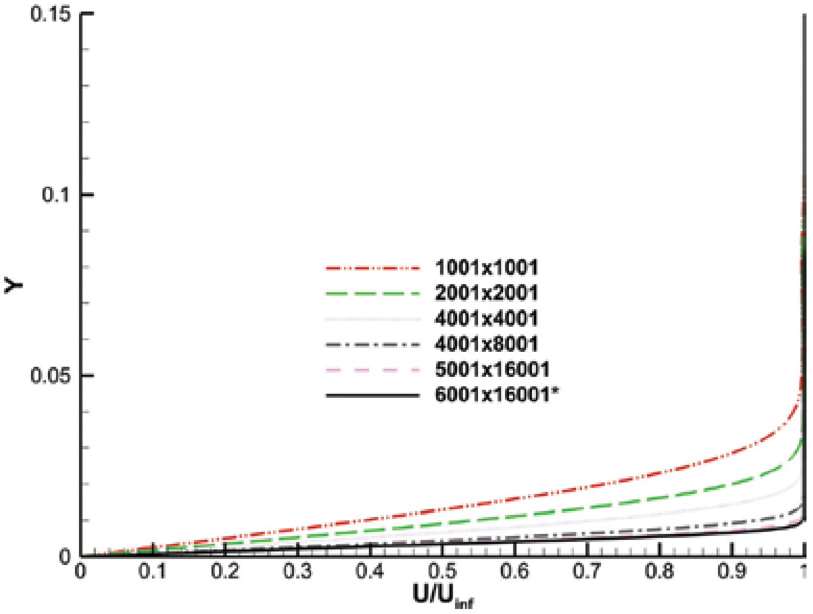


Figure 4. U velocity profile at 0.5*L.

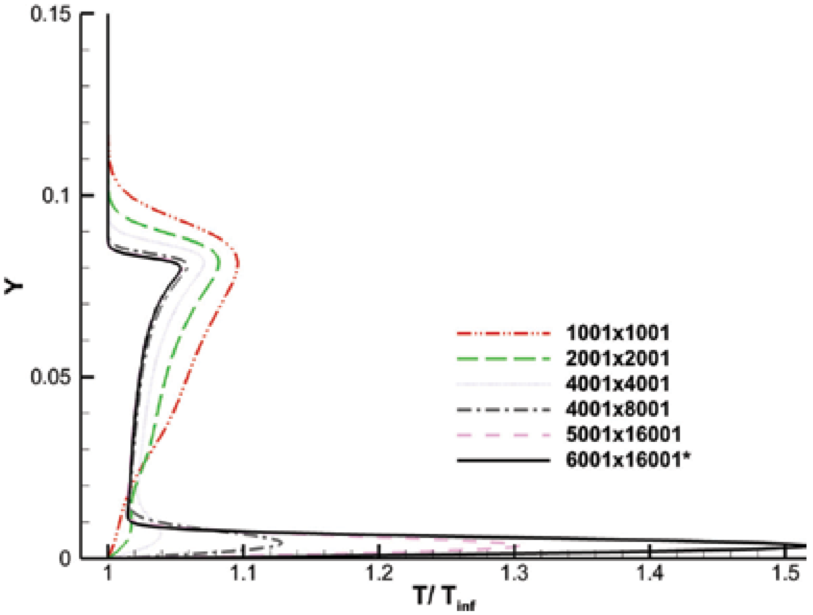


Figure 5. Temperature profile at 0.5*L.

observation of the data demonstrates that the height of the boundary layer is approximately 0.0088 of non-dimensional units, representing 8.8 mm based upon $0.99U_\infty$. In addition, no evidence of shock is shown in **Figure 4**. Similarly, **Figure 5** depicts the temperature profile at

0.5 m from the leading-edge. However, in this case, the effect of viscous dissipation within the boundary layer is clearly demonstrated [10]. As noted in **Figure 5**, the temperature increases from the outer edge of the boundary layer toward the wall, reaching two peaks. The outer peak indicates the presence of a shock, while the inner peak indicates the effects of boundary layer dissipation due to viscous friction. Similar trends were also found in [11, 12].

The grid study data suggest that grid independence was obtained for a grid size of 5001 by 16001. A closer observation of **Figure 5**, reveals the existence of the two expected sub-layers; namely the *entropy* layer and the *viscous boundary* layer. As supported by **Figure 5**, although the height of the shock wave was fully resolved with mesh sizes; 4001 by 8001; 5001 by 16001 and 10001 by 16001*, the dissipation effects were clearly not. Herein, the conclusion is the boundary layer needs an extremely finer set of grids to resolve its physics when compared to the mesh size needed to resolve the entropy layer and the shock wave.

In this chapter, the normal Mach number is measured in the direction of the pressure gradient, and computed with the aid of Eq. (23), which is fully described in [13]. Since pressure and density are the two variables that produce the greatest change as they traverse discontinues in the flow field, they are also very efficient in detecting shocks. In some cases, however, false indications may occur, so a small degree of filtering is required [14]. The filtering criteria proposed by [14] with a threshold of $\epsilon = 0.007$ is used in Eq. (23).

$$Ma_n = \frac{Ma \cdot \nabla p}{|\nabla p|} = 1 \tag{23}$$

Consider the IDS flow field distribution of the normal Mach number computed from expression (23) illustrated in **Figure 6**. Using a filtering threshold of $\epsilon = 0.007$, the bow shock, the viscous boundary layer, entropy layer and other flow field features are extracted. It can be observed that the bow shock starts slightly ahead of the leading-edge tip and it certainly displays the characteristic curvature. This characteristic was also reported in [1]. In addition, **Figure 6** accurately predicts the growth of the boundary layer. More importantly, the similarities of the predicted features illustrated in **Figure 1** closely matches the IDS computed results illustrated in **Figure 6**. Of greater significance is the fact that the two interaction zones; namely the *strong* and *weak* inviscid-viscous interactions zones merged, and all are vividly computed.

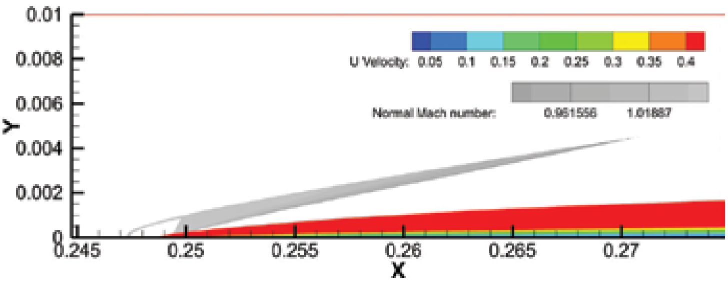


Figure 6. Normal Mach number.

Now, consider the Q-criterion introduced by [15] and computed with the aid of expression,

$$Q = \frac{1}{2}\left(\|\Omega\|^2 - \|S\|^2\right) \tag{24}$$

Note that the symbols $\|\Omega\|$ and $\|S\|$ in Eq. (24) represent the Euclidean norm of the vorticity and rate of strain tensor, respectively. Further, Eq. (24) is an effective tool for the extraction of the vector field topology that represents the local balance between the rate of shear strain and vorticity. When Eq. (24) is applied to the IDS flow field solution over the flat plate, the regions with dominant rates of strains and vorticity are revealed. The Q-criterion results are documented in **Figure 7**. Again, as observed the bow shock, the viscous boundary layer, entropy layer and other flow field features defined by the vortical structures are effectively captured.

In efforts to closely observe the flow physics in the *weak* interaction regime, the variation of the Q-criterion in the y-direction is plotted and illustrated in **Figure 8**. Note, **Figure 8** provides quantitative information about Q-criterion at the location x = 0.27 along the plate and merely complements the information already presented in **Figure 7**. Nevertheless, **Figure 8** reveals that the strain dominates over vorticity very near the wall; as the Q-criterion becomes negative as it approaches the wall. The Q-criterion behavior observed in this analysis is typical within viscous sub layers where the shear stress is laminar [16]. A second layer, the so-called turbulent layer where the swirling motions are common causes the Q-criterion to turn positive. In this region, the viscous effects contribute to large increases in entropy, and consequently vorticity [17]. Thus, this layer is characterized by positive values of Q-criterion. Moving deeper into the flow field, vertically above the wall, the inviscid region is revealed. In this region, the rate of strain dominates over the rate of rotation, and the Q-criterion gets deeper in the negative direction, only to be reversed as the shock wave is penetrated.

A close-up investigation of the flow physics at the hypersonic leading edge was conducted. To this end, an extra case was analyzed where the length and height of the domain were reduced, while the dimensionless parameter, such as Reynolds and Mach numbers were kept constant. Boundary conditions and the freestream values of the primitive variables were the same from the previous analyses. However, unlike the previous solution, where the full plate of length 1 m was studied, the following results do not consider trailing edge. The focus of this analysis

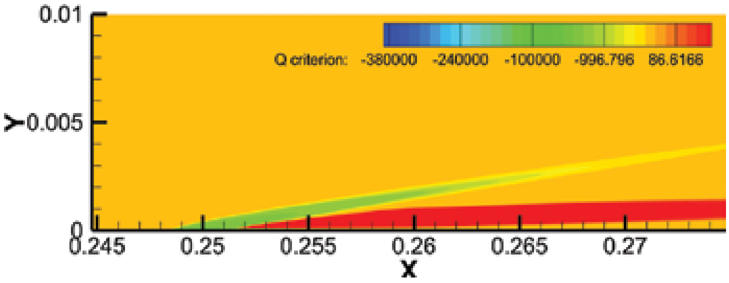


Figure 7. Q-criterion contour plot.

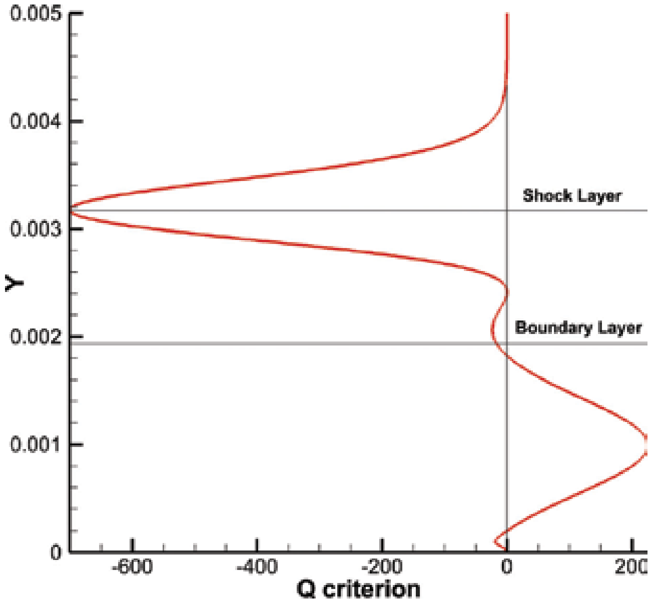


Figure 8. Q-criterion profile at 0.27 from the leading edge.

is to analyze the flow physics near the leading edge thoroughly. The length and height were selected as 0.1 and 0.02 m, respectively and the leading-edge gap was defined as 0.1 dimensionless units. **Figure 9** illustrates IDS prediction in the form of the Q-Criterion at the leading edge. These results indicate that the rate of strain is the dominant effect at the leading-edge followed by large rotational motions that cause a delay in the growth of the boundary layer. **Figure 9** also shows that the bow shock wave is formed ahead of the plate.

It is of interest to note that at the tip of the plate, the region with the greatest rate of strain within the flowfield is observed, albeit a small region. Immediately following this region there is a similarly small region with the greatest rate of rotation, refer to **Figure 10**. The two regions with the greatest rate of strain and rate of rotation are located at the leading edge, and it is from these two regions that the shock wave and the boundary layer respectively emanate.

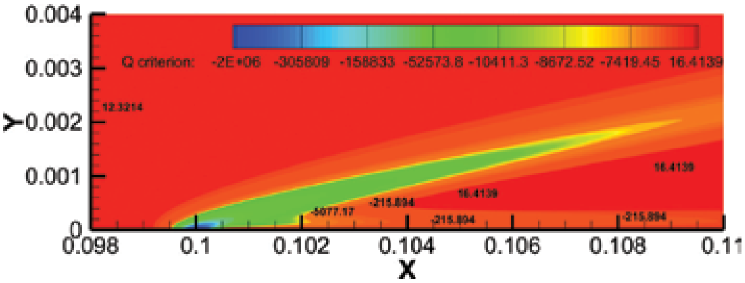


Figure 9. Q-criterion contour plot at the leading edge.

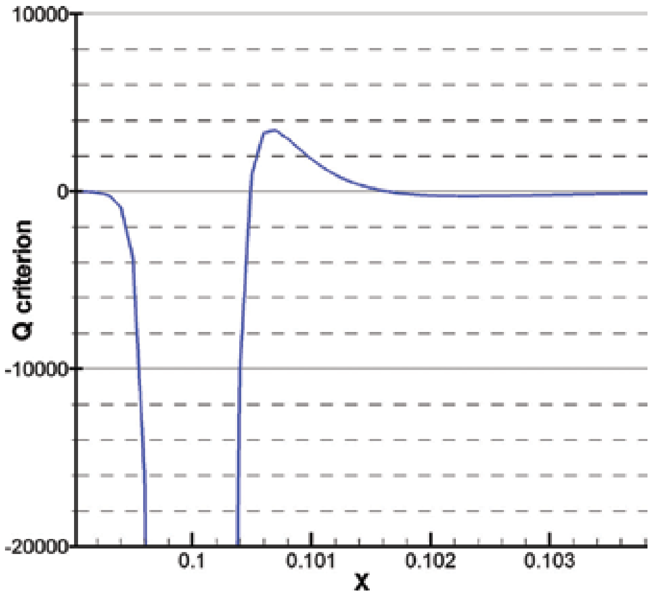


Figure 10. Horizontal profile of Q-criterion at y = 2 × 10^{-5}.

At this point, the technical details associated with the growth of these sub-layers is not fully resolved, and as such, further analysis is warranted.

5.1. Validating the IDS hypersonic leading-edge solution

The experimental data developed by [9] was used to validate the accuracy of the IDS solution to the Hypersonic flat plate problem. In **Figure 11**, the temperature and u-velocity profiles at 80 mm from the leading edge for both cases: experimental and IDS, respectively, are compared. Note that **Figure 11** shows that the IDS solution under-predicts the maximum temperature inside the boundary layer according to the experimental data. **Figure 11** also shows the comparative behavior of the horizontal component of the velocity vector for the experimental study and the IDS solution.

Obviously, there are differences in the experimental and IDS solutions. The reasons for these differences were analyzed. First, it turns out that the required freestream conditions used during the experiment were not public and could not be easily reproduced. Reference [9] described that the study was carried out in a divergence nozzle, and as such, the flat plate experienced a somewhat favorable pressure gradient during the course of the experiment. Second, the freestream conditions were computed using a computational package called STUBE [18], that used the piston theory to estimate the pressure in the reservoir; a huge approximation. Other discrepancies were found between the experimental data and the STUBE predictions [9]. Moreover, [9] also compared the experimental data with the numerical solution from a commercial CFD package called CFD-FASTRAN. The CFD simulations used the freestream conditions calculated by STUBE, and even then, the freestream velocity was

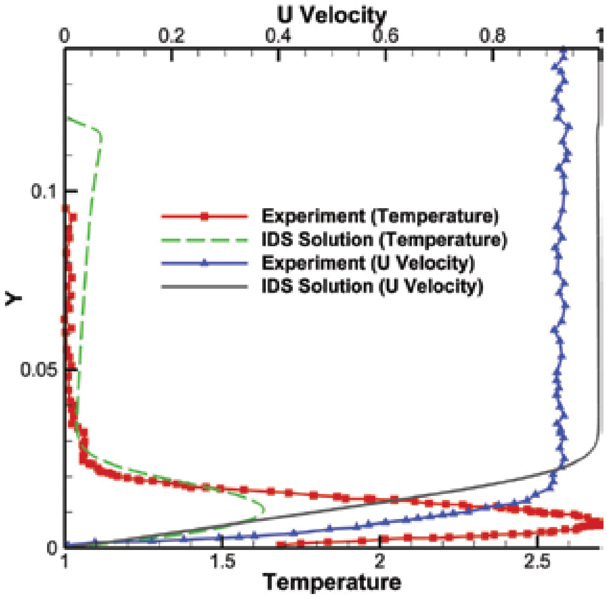


Figure 11. Experimental and IDS solution.

scaled to match the measured external velocity [9]. Nevertheless, even with these uncertainties, the data presented in **Figure 5** is considered to be reasonably close and the IDS code is considered validated.

Figure 12 illustrates the vertical profiles of the density and the y-component of the velocity vector at 80 mm from the leading edge, respectively. However, unlike in the case of the temperature and U velocity profiles, there are no equivalent experimental data available for direct comparison. However, as can be observed in **Figure 12** the three most important phenomena, namely the boundary layer, the entropy layer and the shock wave along with their respective flow filed characteristics are distinctly captured.

5.2. Approximate boundary layer analysis

The boundary layer thickness in the absence of adverse pressure gradient can be computed as [19]:

$$\frac{\delta}{x} = \left[5.0 + \left(0.2 + 0.9\frac{T_w}{T_{aw}}\right)(\gamma - 1)M_e^2\right]\sqrt{\frac{C_w}{\text{Re}_x}} \tag{25}$$

Where C_w represent the Chapman-Rubesin parameter, Re_x is the Reynolds number at the measurement point and $T_{aw,}$ the adiabatic wall temperature, is given by:

$$T_{aw} = T_e\left(1 + \sqrt{\Pr}\left(\frac{\gamma - 1}{2}\right)M_e^2\right) \tag{26}$$

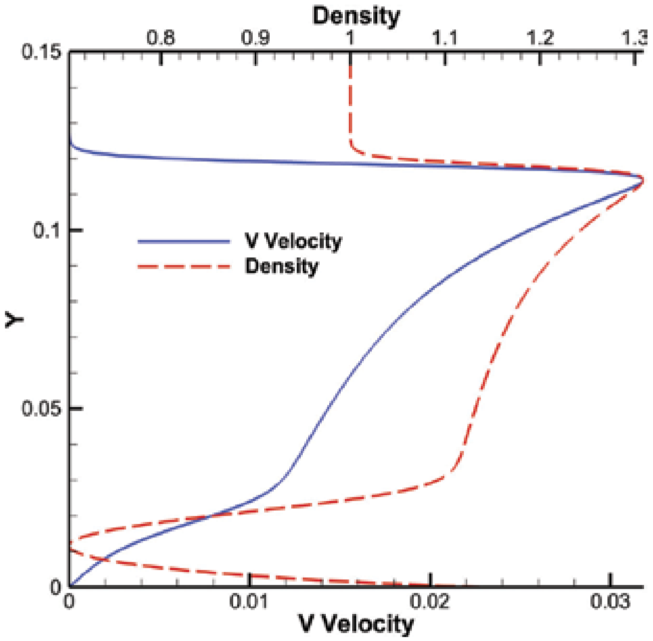


Figure 12. Density and vertical velocity profile.

Equations (25) and (26) were used to compute the theoretical boundary layer, whose value is 1.87 mm. The boundary layer predicted by the IDS was measured to be 2.83 mm. The authors are still evaluating the reasons for the resulting discrepancy.

5.3. Investigating the strength of the hypersonic leading-edge interaction

An important phenomenon that is observed within the flow field in the vicinity of the leading edge is a rise in pressure. This high pressure emanates at the edge and it leads to an induced pressure gradient along the rest of plate. Therefore, the assumption of zero pressure gradient condition through the boundary conditions is debatable [10]. Today's experiments have shown that under hypersonic flow conditions, the leading-edge experiences a bow shock, a significant pressure rise, very large skin friction, and very large heat flux. Further, these experimental observations have shown that adverse conditions are confined only to the leading-edge. To fully explore the leading edge behavior, researchers [19] introduced the shock interaction parameter, χ, which is defined as:

$$\chi = Ma_{\infty}^{3}\left(\frac{C_w}{\mathrm{Re}_{\infty}}\right) \tag{27}$$

This parameter, χ, is a function of the freestream Mach number, Reynolds number and the specific heats ratio, and it serves to quantify the strength of the flow field interaction at the leading edge. Two sets of the shock interaction parameter, χ, ranges are of interest to this study; weak interactions as described by $\chi << 1$ and strong interactions as described by

$\chi >> 1$. **Figure 13** shows the experimental data from [20] indicating the strength of the shock interactions, χ at Mach numbers ranging from 5 to 10 to 20, as indicated by the green, blue and cyan colors. It is important to mention that the horizontal axis is described by the inverse of $A\chi$, where A is an aerodynamic parameter defined by

$$A = \frac{1}{2}(\gamma - 1)0.664(1 + 2.6T_w/T_0) \tag{28}$$

This transformation is done in efforts to allow for small values of $A\chi$ to map with the strong interaction regions and for large values of $A\chi$ to map with the weak interaction regions.

Using the data provided by [20], the weak interaction curve, highlighted in black, is recovered. Finally, the IDS solution is also plotted in **Figure 13** and it is depicted by the red curve. As observed in **Figure 13**, the IDS solution matches the predictions governed by the interaction parameter, $A\chi$, and as noted, lies closest to the Mach 5 interaction curves.

The IDS hypersonic flat-plate solution confirms that the scheme is capable of accurately resolving the complex flow physics within the vicinity of the hypersonic leading edge, and it correctly qualifies the inviscid-viscous interactions associated with this region. However, it is important to note that the IDS scheme is currently being upgraded with OpenMP and MPI capabilities in efforts to handle three-dimensional flows. It is the author's opinion that the mechanism driving turbulence dynamics is three dimensional in nature, and therefore cannot be captured by two-dimensional flow fields. As a result, in 2D flow fields, vortex stretching and other fluid mechanisms important to the development of turbulence flows are absent. Once these capabilities are validated, the hypersonic flat-plate problem will be revisited.

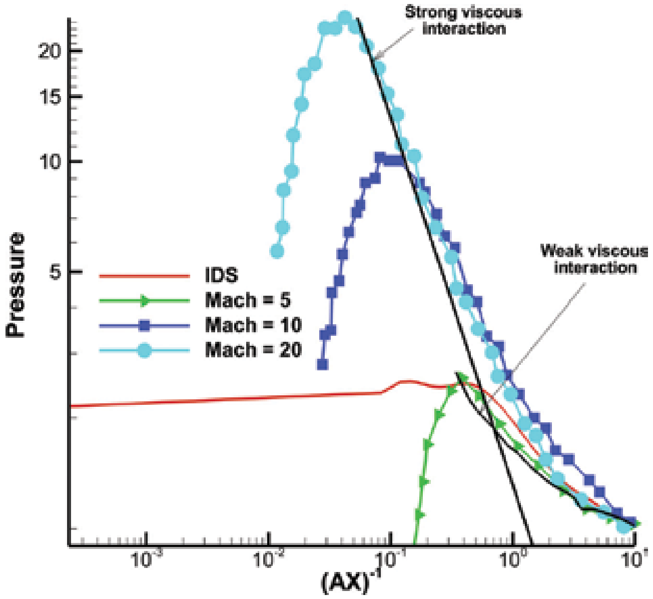


Figure 13. Induced pressure near the leading edge.

Acknowledgements

The authors thank Maida Bermudez Bosch M.A., CCC-SLP for her help on editing and proofreading the written article as well as Professor Kenneth M Flurchick from the Department of physics and Computational Science and Engineering program at North Carolina A&T State University (NCAT) for his guidance and advice on developing the parallel version of IDS. The computational resources used in this research were made available in part by a grant from the National Science Foundation Grant No.#1429464, entitled MRI: Acquisition of CRAY XC-30 HPC Cluster to Support NCAT Research Computing, Education and Outreach. In Addition, the authors want to acknowledge the financial support from AFRL under the Grant/Cooperative Agreement Award #FA8651-17-2-0001.

Author details

Frederick Ferguson*, Julio Mendez and David Dodoo-Amoo

*Address all correspondence to: fferguso@ncat.edu

Mechanical Engineering Department, North Carolina A&T State University, Greensboro, NC, USA

References

[1] Mohling RA. Numerical computation of the hypersonic rarefied flow past the sharp leading edge of a flat plate [Doctoral dissertation]. Iowa State University; 1972

[2] John D, Anderson J. Computational Fluid Dynamics: The Basics with Applications. New York: McGraw-Hill Education; 1995

[3] Blottner F. Finite difference methods of solution of the boundary-layer equations. AIAA Journal. 1970;**8**(2):193-205

[4] Butler TD. Numerical solutions of hypersonic sharp-leading-edge flows. The Physics of Fluids. 1967;**10**(6):1205-1215

[5] Olynick DR, Taylor JC, Hassan H. Comparisons between Monte Carlo methods and Navier-Stokes equations for re-entry flows. Journal of Thermophysics and Heat Transfer. 1994;**8**(2):251-258

[6] Lofthouse AJ, Boyd ID. Hypersonic flow over a flat plate: CFD comparison with experiment. In: 47th AIAA Aerospace Sciences Meeting Including the New Horizons Forum and Aerospace Exposition. Aerospace Sciences Meetings. American Institute of Aeronautics and Astronautics; 2009

[7] Lofthouse AJ, Scalabrin LC, Boyd ID. Velocity slip and temperature jump in hypersonic aerothermodynamics. Journal of Thermophysics and Heat Transfer. 2008;**22**(1):38

[8] Elamin GA. The integral-differential scheme (IDS): A new CFD solver for the system of the Navier-Stokes equations with applications [Doctoral dissertation]. North Carolina A&T State University; 2008

[9] O'Byrne S. Hypersonic Laminar Boundary Layers and Near-Wake Flows [Doctoral dissertation]. Australian National University; 2002

[10] Anderson JD. Hypersonic and High Temperature Gas Dynamics. Virginia: American Institute of Aeronautics and Astronautics; 2000

[11] Anderson JD. Modern Compressible Flow: With Historical Perspective. New York: McGraw Hill Higher Education; 2003

[12] Van Driest ER. Turbulent boundary layer in compressible fluids. Journal of Spacecraft and Rockets. 2003;**40**(6):1012-1028

[13] Lovely D, Haimes R. Shock detection from computational fluid dynamics results. In: 14th Computational Fluid Dynamics Conference. Fluid Dynamics and Co-located Conferences. American Institute of Aeronautics and Astronautics; 1999

[14] Wu Z, Xu Y, Wang W, Hu R. Review of shock wave detection method in CFD post-processing. Chinese Journal of Aeronautics. 2013;**26**(3):501-513

[15] Hunt JC, Wray AA, Moin P. Eddies, Streams, and Convergence Zones in Turbulent Flows. Standford University Center for Turbulence Research; 1988. pp. 193-208

[16] Davidson P. Turbulence: An Introduction for Scientists and Engineers. USA: Oxford University Press; 2015

[17] Babinsky H, Harvey JK. Shock Wave-Boundary-Layer Interactions. New York: Cambridge University Press; 2011

[18] Vardavas I. Modelling reactive gas flows within shock tunnels. Australian Journal of Physics. 1984;**37**(2):157-178

[19] White FM, Corfield I. Viscous Fluid Flow. Boston: McGraw-Hill Higher Education; 2006

[20] Stollery J. Viscous Interaction Effects on Re-Entry Aerothermodynamics: Theory and Experimental Results. In: Aerodynamic Problems of Hypersonic Vehicles. AGARD Lecture Series 42. Vol. 1; 1972. pp. 10-1-10-28

Chapter 5

Smart Community Wireless Platforms: Costs, Benefits, Drawbacks, Risks

Sakir Yucel

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.73838

Abstract

A wireless network covering most of the city is a key component of a smart city. Although the wireless network offers many benefits, a key issue is the costs associated with laying out the infrastructure and services, making the bandwidth available and maintaining the services. We believe community involvement is important in building city-wide wireless networks. Indeed, many community wireless networks have been successful. Could the city inspire and assist the communities with building their wireless networks, and then unite them for a city-wide wireless network? We address the first question by presenting a model where municipality, communities and smart utility providers work together to create a platform, smart community wireless platform, for a community where platform sides work together toward achieving smart community objectives. One challenge is to estimate the total cost, benefits and drawbacks of such platforms. Another challenge is to model risks and mitigation plans for their success. We examine relevant dynamics in measuring the total cost, benefits, drawbacks and risks of smart community wireless platforms and develop models for estimating their success under various scenarios. To develop models, we use an intelligence framework that incorporates systems dynamics modelling with statistical, economical and machine learning methods.

Keywords: smart community wireless platform, cost of community wireless networks, benefits of community wireless networks, smart community, system dynamics modeling

1. Introduction

A smart city aims to embed digital technology across city functions including its economy, mobility, environment, people, living and, governance. Many cities have taken initiatives toward becoming a smart city to foster commercial and cultural development. A wireless network (e.g. Wi-Fi network) covering most of the city is a significant contributor and a major step toward becoming a smart city. Such a network offers many benefits in tackling challenges such as reducing traffic congestion, reducing crime, fostering regional economic growth, managing the effects of climate, and improving delivery of city services [1].

City-wide wireless networks are still desired even with the availability of cellular networks, mainly due to their low cost and higher bandwidth compared to cellular networks. Plus, people are more inclined to use the wireless networks where available as opposed to using their limited data plans. In addition to citizens, many smart IoT devices will require bandwidth and many of them will use protocols which are best supported by a city-wide wireless network.

One major issue with city-wide wireless network is the high cost of laying out the infrastructure, rolling out the services, allocating adequate bandwidth, maintaining the services. One question is who will setup the network and who will pay for it. A second question is who will supply the bandwidth while broadband bandwidth is still in shortage in most cities. Another question is who will pay for the supplied bandwidth.

What should the cities do? Should they rely solely on the wireless operators to build a wireless network across the city? In general, it is unreasonable to expect the private sector to setup a wireless network for smart city objectives. If not private sector alone, then how about some private-public partnerships? Despite numerous attempts in prior years, private-public partnerships and joint ventures between municipalities and private companies have failed to take hold. Furthermore, several states have enacted legislation to prevent municipalities from offering wireless services in many forms in the city for variety of reasons [2]. While there are so many failures in the past and there is political controversy, why should they still pursue a city-wide wireless network? Should they simply give up on their goals of being a smart city? How could they maintain their competitiveness without a wireless network in the digital age?

We think that cities could look for new approaches in realizing city-wide wireless network. One approach is to analyze the success of community wireless networks and try to find ways to leverage their success for building a city-wide wireless network. Indeed, there are many examples of successful community wireless network implementations [3, 4]. In this paper, we will bring attention to successes of community wireless networks and develop a model where municipality, communities and smart utility providers work together to create a platform, which we call smart community wireless platform, where different platform sides work together toward achieving smart community objectives. The purpose for this investigation is to take a new look at building a city-wide wireless network through a new model based on integrating smart community wireless networks over the span of the city. The objective of this paper is to present this platform and its various dynamics. Accordingly, the paper takes more of a conceptual approach rather than a technical one. The purpose is to introduce the smart community wireless platform. How such platforms could join to form a larger city-wide wireless network is a separate discussion we address in [5].

2. Smart community wireless platform

Earlier approaches involving municipality partnering with private commercial providers failed for variety of reasons [2, 6, 7]. We believe smart city starts with smart communities and hence the community involvement is significant in building a city-wide wireless network. We use the term smart to indicate the human factor in building and using the wireless network. Indeed, many community and neighborhood wireless models have been successful. The proposal is take it further through collaborations of communities, municipalities and other partners to realize a city-wide wireless network. In our approach, we will include the smart utility providers as a player in the platform. The question becomes: can communities, the municipality and smart utility providers work together to build a platform for the community? Let us first define the platform.

2.1. Platform definition

We will describe the platform by explaining its system architecture and by discussing its sponsor, its providers, sides, economic utilities and network externalities, financial resources and policies. We will outline the strategies for how to position, present, realize and operate it. For a general discussion of platforms and platform businesses, see [8].

A smart community wireless platform is a community wireless network built and maintained through collaboration of the community, the municipality and the smart service providers. There are multiple sides on this platform: (1) users who use the wireless network and they may also sponsor bandwidth, (2) bandwidth sponsors who sponsor bandwidth for this wireless network, particularly the businesses, (3) other smart service providers.

In this model, the community and municipality assume the main roles. The amount of involvement varies in realizations of this model in different cities and even within different communities in the same city. Municipality plays an important role in this platform in both supporting the communities and organizing them to participate into the city-wide bigger wireless network.

The users are community members or visitors that use the wireless network.

Bandwidth sponsors are entities that sponsor bandwidth used by the users to connect to the Internet. Community members may be users and bandwidth sponsors at the same time. Businesses, non-profits and organizations in the community become bandwidth sponsors. Smart service providers may become bandwidth sponsors. In this paper, we will focus more on businesses as the bandwidth sponsors that provide bandwidth for the users.

Smart service providers offer smart services to the users of the platform. One typical example is the utility provider companies like electricity, gas, water, waste management. For example, waste management company provides services for smart garbage collection to the users of the platform. We will not use the term utility for them (as in electricity, water, gas) in this paper as we will use the term utility to refer to economical utility for being on the platform. We will call them as smart service providers. These smart service providers use the community wireless network for communication of their smart devices (sensors, smart devices, and other IoT devices) that they place in the network. They benefit from the platform by placing IoT devices that use the wireless network for communication, or more likely by building sensor networks that integrate with the wireless network. They sponsor bandwidth so they become bandwidth sponsors and they may provide other components into the platform as explained in later sections.

Another example to smart service providers is the city offices and department. For example, parks department provides services for park resources. Another example is the community itself in providing smart services to its members, for example, smart education services.

2.2. System architecture

Figure 1 shows the layered system architecture for the platform. It has the following layers:

Smart Community Wireless Network Layer: Wireless network is built on top of the wireline infrastructure that has network nodes, servers and cabling provided by community, municipality and smart service providers.

Middleware, Data and Infra Layer: Computing and storage infrastructures belonging to the community, municipality, and smart service providers in this system store data and offer computing, networking, caching and data storage resources. It contains software platforms and services including middleware, service oriented solutions, fog and cloud computing infrastructures (both commercial and community clouds). Other components include reliability, security, privacy and trust solutions. Infrastructure offered by smart service providers hosts the data and resources for smart services. Those infrastructures could be accessible through the wireless network. Given the complexity of the smart services and the extent of the data generated by the platform sides, this layer should be capable of storing and processing the data. It should be able to store and process various types of data including events,

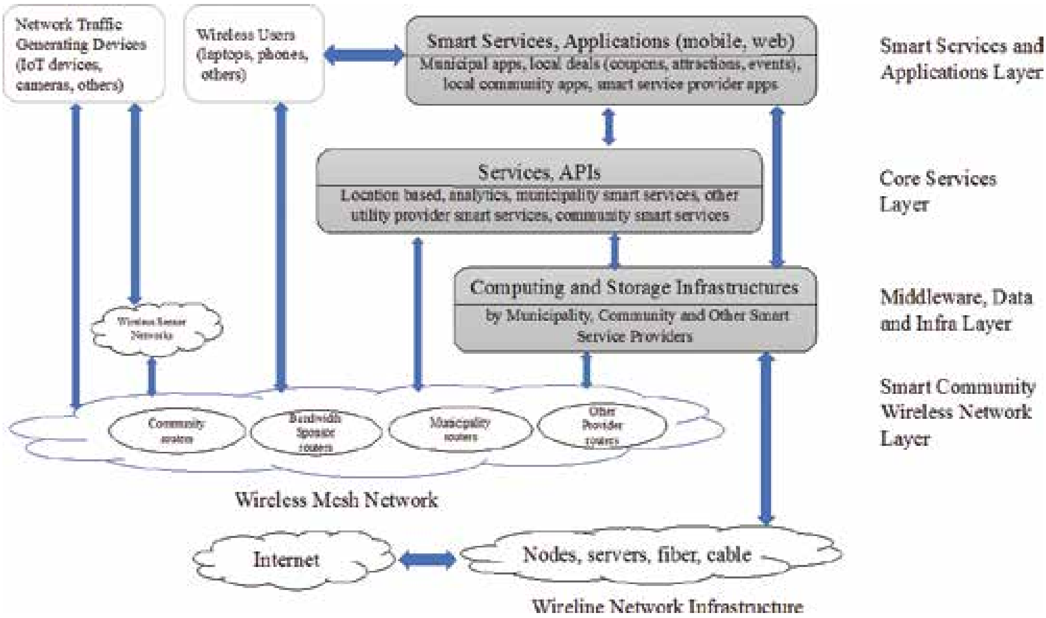

Network Traffic Generating Devices (IoT devices, cameras, others)
Wireless Users (laptops, phones, others)
Smart Services, Applications (mobile, web)
Municipal apps, local deals (coupons, attractions, events), local community apps, smart service provider apps
Smart Services and Applications Layer
Services, APIs
Location based, analytics, municipality smart services, other utility provider smart services, community smart services
Core Services Layer
Computing and Storage Infrastructures
by Municipality, Community and Other Smart Service Providers
Middleware, Data and Infra Layer
Wireless Sensor Networks
Community routers
Bandwidth Sponsor routers
Municipality routers
Other Provider routers
Smart Community Wireless Network Layer
Wireless Mesh Network
Internet
Nodes, servers, fiber, cable
Wireline Network Infrastructure

Figure 1. System architecture for smart community wireless platform.

unstructured, structured, geo-spatial, crowdsourcing data. The smart service provider infrastructures may be required to perform real-time processing.

Core Services Layer: Some core services such as analytics, user location tracking, location-based services, search, semantics, context processing, visualization, collaboration platforms, geo-spatial services, access to public data and statistics, community social networking and other digital community services are offered using this infrastructure which is accessible by the wireless network. Outputs of data analytics and mining are offered at this layer. These core services are usually accessible to bandwidth sponsors and smart service providers.

Smart Services and Applications Layer: This layer includes all smart services offered by the smart service providers such as smart transportation, smart health and smart government services. This layer offers APIs of the smart services for application builders. Additionally, it includes mobile and web user applications offered by municipality, community and smart service providers.

Wireless users and IoT devices use the wireless network and generate network traffic. IoT devices belong to the smart service providers that participate in this platform. They are usually part of wireless sensor networks but can be directly connected to the community wireless networks. Network traffic from some of the IoT devices would not leave to the Internet, but rather stored, processed and analyzed in the infrastructure accessible by the community wireless network.

With this architecture, various services can be offered in a modern smart community. All these services are provided by smart service providers (e.g. private companies, municipal offices and communities). The platform creates an ecosystem around this system architecture. This architecture hosts an internet of everything environment including connected devices, users, communities offering community services, smart service providers using the network and offering their smart services.

The smart service providers are vertically integrated to provide their services over the platform. Some of these services are available only on this platform. The providers use the wireless network for collecting data from field devices into their infrastructures and may offer the same services over the Internet, which can be accessed by anyone. In any case, the wireless platform provides a home for the devices and for collecting their data. The providers contribute to this network by supplying access point routers that connect their devices to the wireless network and preferably by sponsoring additional bandwidth.

Internet access is supplied by bandwidth sponsors and commercial ISP services.

We will not present a detailed design of the wireless network in this paper, rather we will state our assumption. We assume the platform uses a mesh Wi-Fi technology as it is most often the technology used in such networks. In the network, there are access points and routers supplied by the community usually having generic server hardware and running open source firmware and software. The mesh network usually runs open source mesh network routing software and open source network management software to setup and manage a software-defined wireless mesh network. In addition to routers supplied by the community, there are routers supplied by the community members and other routers belonging to municipality, to the sponsors of bandwidth and to the smart service providers. The design should cover the whole target area by adding intermediary routers in places where no sponsor router is available and should be able to redirect user traffic to any of the available access points.

The platform relies on community members, businesses and organizations to share a portion of the total required bandwidth to access the Internet. So, a significant assumption is that ISPs allow plan sharing in their service terms. When bandwidth sharing does not supply the required bandwidth completely, remaining bandwidth needs to be purchased from the local commercial ISPs.

The mesh Wi-Fi network uses mechanisms for access control, metering and blocking the user traffic beyond a daily cap. It enforces rate limiting of the users with respect to data rate and the amount of download/upload. It employs self-adjusting network functionalities for fairness such as enforcing dynamic rate limiting the bandwidth to each wireless interface based on the current total number of users. When the number of users exceeds the network capacity based on minimum bandwidth for each device, new connection requests are not granted thanks to dynamic connection admission control. Therefore, some users will be blocked and not able to join. The city officials and municipal services have guaranteed service for accessing the wireless network, and they are not blocked.

The wireless network should offer enough bandwidth to fulfill the basic requirements of the users and support applications that will benefit the community and the city. Such applications include community social networking, community calendar of events and information about events, services offered by community, municipality and commercial smart service provider's offer. On the other hand, it should not be positioned as a competitor to commercial cellular or wireless networks as we argue in Strategies For Platform Promotion and Positioning section. For example, it should not allow unlimited upload and download. One option is to rate limit the download/upload speeds. Another option is to limit the traffic to and from the Internet while the users could enjoy unlimited access to the smart services. In other words, their Internet traffic is metered and capped, however, traffic within the wireless network could be unlimited, or limited with a higher cap subject to the whole capacity of the wireless network.

The wireless network implements typical security and access control mechanisms [9, 10].

When similar networks are integrated together, a seamless network covering a bigger span of the city could be possible [5].

2.3. Platform control

For the platform, it makes sense for the community to be the platform sponsor and the primary provider. Community has the say on management and policies of the network. The community decides on what policies and what strategies to apply. In another arrangement model, municipality and community may behave as the platform sponsors, but we will assume the community is the main sponsor and provider of this platform in this paper.

We assume no commercial offerings using this platform by municipality due to existing state laws. We assume the community does not engage in seeking any profit using the platform. The platform is not commercial and is not for profit for the community. For this reason, many of the concerns applicable to commercial platforms do not apply, like pricing but some other concerns apply like funding. We assume the platform will be free for users but possibly with some volunteering or sponsoring bandwidth in return, or with agreeing the usage terms and giving up some privacy.

The term sponsor is used in two meanings in this paper and they should not be confused: one in the meaning of platform sponsor which is the entities that control the platform. The other one is the bandwidth sponsor, for example, businesses that provide bandwidth for the wireless users to connect to the Internet.

Community mainly supplies volunteers and bandwidth sponsors. Municipality helps by allowing the community to use city owned light posts, traffic lights and municipal buildings for attaching access points, and by allowing to use wireline infrastructure; assisting the grant writers with grant applications; providing access to GIS mapping data; assisting with network design; financial help by identifying grants and tax breaks for community networks. With community and municipality working together, businesses and other smart service providers would join the bandwagon increasing the network externalities and adding value for the platform, therefore making it a viable platform.

The openness of the community wireless network is controlled by the community. Normally, the platform is open to any user provided they accept the usage policies. The community will decide on the criteria for who can join as developer and providers of services and on what conditions. The community decides whether the wireless network and infrastructures are open to any developer to develop some service/application, or to any smart service provider to install devices and provide services. The community will decide if research tools can be deployed by universities, or by local startup companies. The community controls the quality of the wireless network and the services offered on it. The community decides and governs what complements such as location tracking and analytics can be provided and by whom. The community makes these decisions following their decision making methodology, for example, may perform SWOT analysis for the complement providers and smart service providers. The complement providers could be commercial ISPs to sell bandwidth, other providers of the equipment, software, services and know-how.

2.4. Utilities and network effects

Each side of the platform has an intrinsic utility for being on the platform. For example, users have utility with being on this platform in the form of getting wireless service and additionally by receiving coupons, deals and other location-based offers, and accessing smart services. Businesses have utility for promoting and advertising their businesses. Similarly for the municipality and other smart service providers.

In addition to the intrinsic utility, each side experiences additional utility due to network externalities. Network effects exist impacting the utility of different sides for being on this platform. It is assumed that the user utility and network size would follow a logistic ("S"-shaped) function. Same side effects will help increase the user population thanks to information diffusion initially. Increase in population would later negatively impact the utility due to congestion, possible degradation in the quality of the wireless network and being blocked in the shortage of available bandwidth.

Cross side network effects exist. There could be positive network externality between users and bandwidth sponsors. Similarly between users and smart service providers. Policies and strategies would increase the network effects and thereby the utility of different sides. We will discuss some policies and strategies for taking advantage of the network externalities in subsequent sections. Network effects are so many and will be outlined in later sections.

2.5. Platform evolution

The community should exercise policies that will allow users, community members, component providers to offer ideas and contributions. The platform evolves by being open to community needs, fostering innovation by allowing community startups and pilot projects and university research and being a testbed for innovation. The use of open source supports such collaboration between global developers and the community developers. Collaboration among communities and tracking what other communities do will help evolve the platform into new technologies and approaches. The community should be transparent to the users about the evolution of the platform as to provide more accurate information about the future roadmap and shape the user expectations accordingly.

The community needs to monitor the total payoff and estimate the lifetime customer value (LCV) and a new user's impact on existing users' utility. Based on these, the community should find the optimum number of users the wireless network should support before considering investment for updates, upgrades and expansions. Therefore, there will be blocked users.

2.6. Financial resources

The smart community wireless platform reduces the financial responsibility for the city, but funding is still required. The municipality could continuously track and search for funding and try to maximize the amount of funding collected for the community platforms. For each grant, the municipality could maintain the area, the purpose, the conditions and constraints of the grant. Certain grants are given for specific applications and purposes (e.g. safety, energy, climate preparedness, transportation, health) and by different sources (e.g. by the Department of Homeland Security, Department of Transportation, Department of Energy, Department of Commerce, and the Environmental Protection Agency). Although many funding opportunities and sources exist, different communities may focus on different ones based on how the opportunities fit their needs and objectives. The community also should be on the look for funding sources by mobilizing its volunteers.

Funding sources include grants, tax benefits, donations (from local organizations, businesses, and community members), bandwidth sponsorships, new exploratory research funds, new hackathon challenges and awards, free services from companies (e.g. free cloud service), testbeds (for trials of different technologies, ideas, models), special funds (e.g. system and service for first responding by department of homeland security), and crowdfunding opportunities [3–5].

2.7. Policies

What should be the ownership, maintenance, and security policy for the platform? It is expected that the community owns the platform as being the sponsor. Regarding maintenance policy, it is expected the community maintains the network with the help of volunteers and part-time contractors.

Who should do the authentication and authorization of users? What should be the user privacy policies? For this platform, the community should control authentication and authorization policies. The users would have to give up on some of their privacy by entering their profile (e.g. via a survey at first login) and agreeing for being tracked for usage and for location. This information is used for analytics and improving the wireless experience, and is integrated into the loyalty programs and deals of the sponsoring businesses. The information is shared with the sponsoring businesses, which is an incentive to bring in more businesses and increasing their utilities. Additionally, this information is used to trace individuals in case of any illegal online activity on the network.

2.8. Strategies for increasing utility of platform sides

Strategies should be developed and employed to increase the value of the platform for different sides and to reach the critical mass in terms of regular users within the community and by visitors. These include strategies to mobilize the volunteers, officials, sponsors, non-profits, smart service providers, commercial ISPs, component and complement providers. There should be strategies in place for:

- Bringing in users by offering them a consistent service as well as access to business loyalty programs and smart services over the platform.
- Convincing private investment for upgrading the broadband infrastructure. Strategies for private investment to lay down more fiber, upgrade wireline infrastructure and services, improve wireless coverage and employ latest technology should be in place.
- Motivating city universities for research and development to help with technical and managerial initiatives. Provide them opportunity to test ideas, provide them with testbeds, nourish their business and management ideas.
- Bringing in other organizations (e.g. non-profits): they will not have much utility for offering their bandwidth just so that more people visit an economic district, but would have increased utility with contributing to the community in developing areas and for reducing digital divide.
- Bringing in smart service providers such as smart parking and waste management.
- Effective crowdsourcing and crowdfunding
- Convincing ISPs to allow sharing the broadband connection.
- Convincing businesses to sponsor by presenting how the platform could lead to more customers, by allowing them to advertise and do directed marketing by accessing the user data, tracking data and analytics on them.
- Convincing smart service providers to sponsor bandwidth in addition to the bandwidth they should provide for their IoT devices.
- Convincing users to sponsor bandwidth: the user is expected to share bandwidth to be able to utilize the network beyond a cap. This option is effective in a residential community, not in a business community where users are mostly visitors.

In residential areas, a crowdsourcing strategy could be employed for residents to join the wireless network and contribute from their broadband connection [3, 15]. The same strategy would not work in a business district. Rather, a strategy that increases utility of the bandwidth sponsoring businesses and non-profits in the area would be more effective.

2.9. Strategies for platform promotion and positioning

Other strategies include positioning the platform, its launch and promotion. What strategy should the municipality follow while helping community mesh wireless and rolling the smart city services on this network, and meanwhile encouraging private investment? Municipality and communities must appreciate the value of commercial investment in the city, should stay away from any policy or strategy that will deter them. The platform should not be positioned to compete against commercial wireless services and substitutable offerings. It should not be about being a winner in the market. Rather, it should be for serving a real need in the community for a specific purpose and to fill the gap from commercial providers. All policies and strategies should be compliant with these principals, that is, keeping the availability of substitutable and commercial offerings. Otherwise, the platform could deter private investment. Additionally, the city may run into legal issues as happened in earlier attempts [2]. Policies and strategies should encourage broadband modernization by private industry both in wireline (fiber) and wireless (5G). On the other hand, community and municipality could try to convince the local incumbent ISPs to lower prices and alter terms of service agreements as this happens with community networks by grassroots groups [1]. In our opinion, this platform should not be positioned as an alternative to conventional ISPs in the last mile, rather a balance should be preserved so that commercial ISPs still find interest and profit in the community. As an example for not competing against substitutable offerings, the bandwidth in the community wireless network should be limited to basic use so that commercial providers still find interest in providing better quality services for fee.

3. Intelligence framework

One important question is how much would be the total cost of building and operating such a platform. Another question is how to measure the benefits and drawbacks to estimate the returns on investment over a period. Another question is how to model risks and mitigation plans for the success. One objective of this paper is to examine relevant dynamics in estimating the total cost of these platforms and develop a model for estimating the cost under various conditions and scenarios. Another objective is to examine relevant dynamics in estimating the benefits, drawbacks and risks and develop models for measuring them. We address these objectives by using an intelligence framework.

In our earlier work [11], we used an intelligence framework for analyzing platforms in general. We will use the same intelligence framework for analyzing the smart community wireless platform with some additional analysis and decision making techniques. Our framework incorporates developing system dynamics (SD) models together with use of economical, statistical and machine learning models. SD modeling and statistical methods have been used for analyzing municipal wireless networks in earlier work [2, 6, 7]. SD modeling in general is used for understanding and analyzing business and management related issues such as estimating cost, benefits and return on investment, and risk analysis. To estimate the total cost over a period, simulation is a powerful tool to try out different scenarios. When detailed statistical analysis could not be done due to shortage of data and exact understanding of how the system works, SD models involving hypothesized assumptions can be valuable tools to demonstrate expected impact of various business decisions when there are feedback relationships among the involved dynamics. With SD model, one can also model the network effects among different sides of the platform and model how the economical utilities due to network effects behave. In our intelligence framework, estimation is done by building SD models together with economical, statistical and machine learning models, running simulations and performing sensitivity analysis. Qualitative SD approach improves system understanding and prediction for various scenarios, even in the absence of quantitative data. This framework has been used in [11–14].

Three are many variables used in the intelligence framework. For the cost and benefit models we develop in this paper, we describe the dynamics and their relevant variables in later sections. Application of this framework into various problem domains requires utilizing additional analysis and decision making techniques and approaches. The problem domain of community wireless network design requires us to further employ the following tools and methods into our intelligence framework:

1. The use of GIS data and mapping techniques for asset mapping and geographic area characterizations while planning and designing the wireless network.
2. The use of network design simulators to estimate needed bandwidth based on expected number of users, expected applications and their QoS requirements like the response time, and for simulating the impact of content caching, location tracking, IoT traffic.
3. The use of tools and models for wireless network security analysis and assessment.

In this section, we will outline how GIS mapping and network simulation exercises would be used in the wireless network design which correspond to the first two items above. Security analysis and assessment related data will not be discussed.

3.1. GIS data and techniques

As the platform relies on community sponsors, all potential assets should be identified and mapped using GIS tools. The assets include bandwidth sponsors such as businesses, hospitals, community based organizations, libraries, schools, religious organizations. These should be reached to find out if they would like to participate in the platform as bandwidth sponsors. They could be classified into three groups like large, medium and small bandwidth providers and mapped in different colors in GIS.

The assets also include the city light posts, buildings, municipal facilities and other physical infrastructure for attaching the equipment. Physical characteristics such as terrain, elevation and alternate infrastructure should be taken into consideration in the design as well as trees and buildings that may be barriers to line of sight between the access points.

Given the size of the area, expected population to use the wireless service, expected usage patterns, the bandwidth requirements of different sections in the area could be estimated and marked in GIS in different colors. Based on needed bandwidth in different sections, how many root access points and where to place them could be designed. In general, root access points should be positioned close to large bandwidth sponsors. After placing the root access points, other equipment mainly the mesh access points are placed in GIS. This is done based on assumed range of the mesh access routers and recommended number of mesh access points per root access point. The distance between the nodes might vary based on the distractions in line of sight between nodes. It would be helpful to draw circles around the nodes for representing their coverage.

This exercise helps with estimating the number of root access points and mesh access points. It shows in GIS where enough bandwidth is being sponsored and where additional bandwidth is to be purchased. It helps with estimating how much bandwidth is to be purchased. Additionally, it shows which sections of the area are well covered and which sections do not have enough wireless coverage.

This GIS mapping exercise addresses the usage by users only. The same exercise should be carried out with smart service providers for their IoT devices.

3.2. Network simulation

The network simulation exercise is to help with planning, logical designing, optimizing and reconciling the community wireless network design. The planning phase helps with developing an estimation for usage patterns and network traffic categorization. This exercise is done under the assumptions of the objectives of the community and together with the GIS mapping exercise.

The logical design phase is about creating a logical design that represents the basic building blocks and the structure of the network. This high-level design considers options for how and where to connect the wireless network to the Internet. This is done also together with the GIS mapping exercise. The high-level topology of the network is logically designed for further simulation purposes. A hierarchical model having core, distribution and access layers is common for the high-level topology. The core layer abstracts the wireline infrastructure including backhaul nodes, fixed routers, cabling and dedicated Internet connections. The distribution layer contains the root access points. The access layer contains the mesh access points and the user devices. The logical design is used in simulations to estimate QoS performance and availability measures. This function is complex and requires building network simulation models and running them to compare wireless network and infrastructure performance. However, traffic simulation is worth the effort as it can result in tangible benefits such as studying the need for content caching in the community network to reduce traffic to/from the Internet, estimating the impact of network security features, and modeling the impact of different wireless network policies.

The reconciling phase brings together the simulation results against the objectives and cost constraints. This help the platform providers to re-evaluate the objectives and assumptions on the platform, to re-think about the hypothesis and to more clearly see the inefficiencies or flaws in initial estimations.

This network simulation exercise addresses the usage by users only. The same exercise should be carried out with smart service providers for their IoT devices.

4. Developing models for cost estimation

In this section, we focus on SD models for estimating the initial and maintenance costs.

4.1. Methodology for developing cost models

We suggest a methodology for developing cost models. The methodology suggests:

1. Work out the characteristics of various cost related dynamics which will be needed in simulation and decision making. These dynamics should be elaborated for a specific community platform through the activities of needs assessment, resources analysis, partnership analysis, asset mapping, network/security/operations planning, policy development. These are done best by the community itself, as the community leaders, volunteers, stakeholders are most aware of the needs, resources and capabilities.
2. Once conceptual dynamics are characterized and cost related variables in those dynamics are identified, these dynamics are incorporated into SD models for estimating the cost. Characterization of several dynamics including financial resources, policies, strategies and utilities are outlined in earlier sections. We will characterize additional dynamics together with cost related variables in the subsequent sections.
3. With data collected during planning, development and operational phases of the wireless platform, build and fine tune the statistical, economical and machine learning models, integrate them with SD models, validate and fine tune the SD models.

In this paper, we will apply the first two steps above.

4.2. Community characteristics

The exercise for characterizing the community should yield values for the following:

- How big the community and how many different potential service areas exist
- How much volunteering from community: How big the volunteer groups (for setup, for maintenance activities, for security and customer service requests)
- The amount of community help for finding grants, needs assessments, publicity and promotion, setup and installation, integration
- How successful the crowdsourcing could potentially be in the community
- Community effectiveness for implementing the strategies for bringing in bandwidth sponsors and smart service providers
- Community help for finding sponsorship and its effectiveness for convincing partners

- Availability of technical skills in the community
- Community effectiveness for developing technical solutions for the wireless network

4.3. Service area characteristics

A community is part of the city like a neighborhood. A service area is an area/district within a community. Our assumption is that there could be multiple service areas within a community and each service area could be different. For example, one service area could be a business district with economic development objective, whereas another one could be a residential district with objective of reducing the digital divide. Where the community has different service areas with different characteristics, it makes sense to characterize service areas separately. A community wireless platform becomes the union of possibly several wireless networks in different service areas with different dynamics.

The exercise for characterizing the service area should yield values for the following:

Demographics related: Population move in, move out and growth rates. Population during day, night. Resident population, visitor population.

Businesses related: Number of businesses willing to share bandwidth and how much bandwidth they will share. Social responsibility awareness scale of businesses in the area.

Substitutable offerings related: Availability and quality of cellular services and hotspots.

Setup related: Size of the area. Availability of city light posts, buildings, municipal facilities and other physical infrastructure for attaching the equipment. Existing IT and networking infrastructure like fiber, municipal IT resources, smart service provider resources. Other geographic and dwelling factors (building, roads, rights of ways) that will impact the setup.

Attractiveness related: Service area attractiveness for grants, sponsorships, donations. Attractiveness for the visitors including shops, places, accommodations. Social initiatives and public services that will impact attractiveness.

Usage related: Projected initial usage characteristics and demand: How many residents will use the system? How many visitors will use the system? What percentage of users use how many times, when and how long? What types of digital activities do community members often perform on wireless network? Peak hour characteristics of the usage? What smart services are available in the platform?

4.4. Municipality characteristics

The exercise for characterizing the municipality should provide values for the following:

- The extend of municipality help with allowing to use traffic lights, light posts, municipal buildings in the community. IT infrastructure elements such as cache servers the municipality could provide to the community.

- Municipality help with grant finding and preparing applications to grants
- Municipality help with launch, publicity and promotion
- Municipality help with network design, setup, installation and integration
- Municipality help with ongoing operations: help with security admin
- Municipality help with training people maintaining the network and the volunteers

4.5. Wireless network and infrastructure characteristics

The exercise for characterizing the wireless network and infrastructure should provide values for the following:

- Characterization of the wireless mesh network: total available bandwidth, overall throughput, latency averaged over all APs based on the design of the mesh network. Availability and reliability of the network. Overall security score of the network. The number of APs in the mesh. How many users can be supported at maximum.
- Characterization of the wireline network: similar to that of the wireless mesh network.
- Characterization of the computing and storage infrastructures: latency and throughput for typical use case transactions. Number of transactions per second per use case.

4.6. Online services and applications characteristics

The number, availability and quality of services in the area increase the utility for users and sponsors. We characterize the services in terms of their availability and quality, and bandwidth demands. These are needed for estimating:

1. Projected application and bandwidth characteristics
2. Attractiveness of the service area to visitors

The exercise for characterizing online services and applications should provide values for:

What services offered to users: Online services from the municipality offered in the service area (for safety, security, municipal services). Services provided by smart service providers. Location-based services using wireless network and IoT beacons for coupons and loyalty programs in business districts. Community online services such as community social network, community cloud. Community virtual visitor app that highlights locations, attractions, points of interest, events, local deals.

Availability and quality of the services: Estimated initial values for these and real monitored/observed values when the services are operational.

Bandwidth demand: Expected number of users for offered services in the area, initially estimated but later monitored. How much data traffic is generated within the wireless network. How much traffic is to be transmitted outside of wireless network without using the

Internet but using network infrastructures supplied by municipality or other smart service providers. How much traffic will be transmitted to the Internet. For example, security cameras feed data traffic into wireless network and wireline infrastructure. This data mostly remains within the network and not need to go to the Internet or other networks. On the other hand, vehicle traffic monitoring cameras feed data into wireless network and this data may go to other networks over infrastructure and streamed to the Internet possibly via separate ISP connections.

4.7. Smart service providers characteristics

Smart service providers place sensors and other devices into the wireless network. They use these devices for their own purposes and they also offer smart applications (e.g. waste monitoring). One characterization is to figure out the amount of data traffic their devices will generate within the wireless network: the traffic transmitted over the networking infrastructure till the data reach the local infrastructure of the smart service provider and/or used by the users on the platform, or reaches the private network connection of the smart service provider. The amount of Internet traffic used by their devices.

Another characterization is for finding out the amount of bandwidth they will sponsor. This amount should be equal or higher than the bandwidth generated by their devices and users. There are two types of bandwidth they need to sponsor: one for the wireless network and the infrastructure for data to remain in the network, and the other one for the Internet. For the infrastructure, the smart service provider should contribute APs into the mesh network. For the internet, the smart service provider should sponsor at least enough bandwidth for their own Internet traffic. Smart service provider may have dedicated connection from the wireless network to their data centers for transmitting IoT data. It is assumed that the smart service providers have their own Internet connectivity from their data centers.

4.8. Quality and attractiveness of wireless network characteristics

Initial attractiveness of the wireless network mostly depends on service launch strategy. Ongoing attractiveness depends on:

- How municipality and community promote and advertise the network
- How they incentivize the citizens to use the network
- The quality of the wireless service with respect to QoS (availability, bandwidth for each Wi-Fi interface, throughput, latency)
- Availability and quality of substitutable offerings such as hotspots and cellular services
- Quality of customer service for security and other usage tickets
- How the network evolves in response to changing usage characteristics: an analytic roadmap is needed to monitor the patterns and re-engineer the network accordingly.

5. SD models for cost estimation

In this section, we develop SD models taking into consideration the cost related dynamics and variables. We consider a service area with economic development objectives in a business district. Since our focus in this SD model is economic development for a business district, the model we present may not apply directly for residential areas.

5.1. Initial setup cost estimation

Figure 2 shows a simple linear SD model with no feedback loop for initial cost estimation for a single service area. It is simple and linear as there is no economic utility to calculate nor any network effect to incorporate.

Cost variables include: Needs assessment fixed cost, Cost of needs assessment, Grant application fixed cost, Cost of grant application, Net from the grants, Raising donations fixed cost, Cost of raising donations, Net from donations, Sponsorship raising fixed cost, Cost of raising sponsorship, Cost of launch, publicity and advertisement, Network design cost, Equipment and software cost, Setup and installation cost, Integration cost.

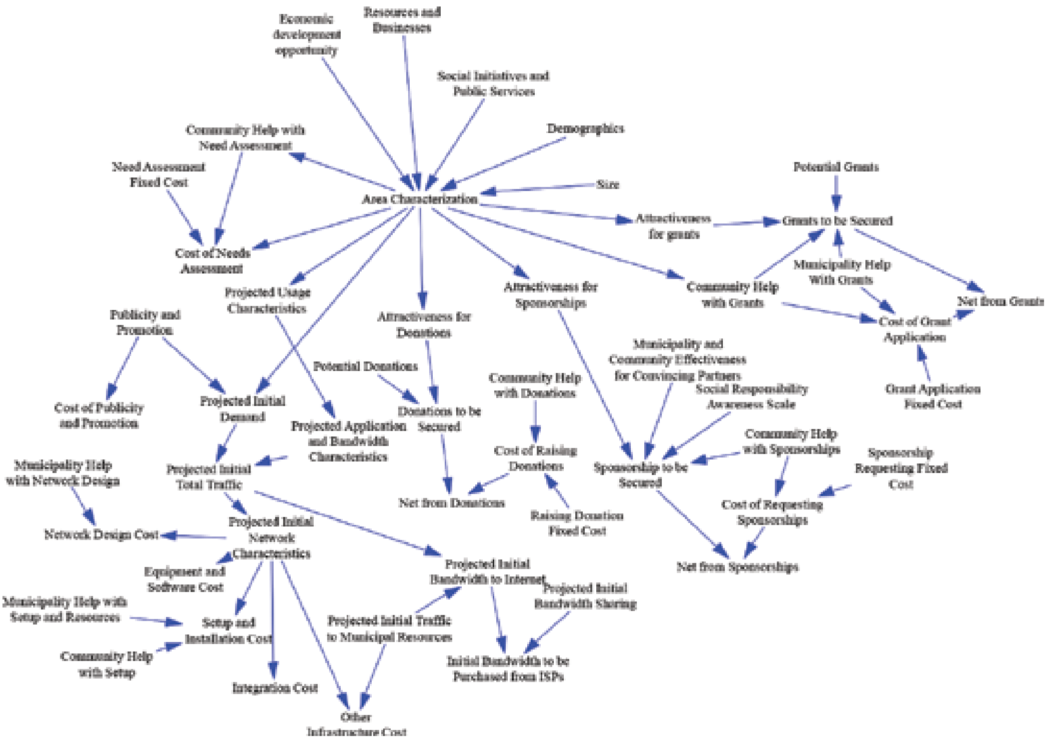

Economic development opportunity
Resources and Businesses
Social Initiatives and Public Services
Community Help with Need Assessment
Demographics
Need Assessment Fixed Cost
Size
Potential Grants
Area Characterization
Attractiveness for grants
Grants to be Secured
Cost of Needs Assessment
Municipality Help With Grants
Projected Usage Characteristics
Attractiveness for Sponsorships
Community Help with Grants
Net from Grants
Publicity and Promotion
Attractiveness for Donations
Cost of Grant Application
Municipality and Community Effectiveness for Convincing Partners
Potential Donations
Community Help with Donations
Social Responsibility Awareness Scale
Grant Application Fixed Cost
Cost of Publicity and Promotion
Projected Initial Demand
Donations to be Secured
Projected Application and Bandwidth Characteristics
Community Help with Sponsorships
Cost of Raising Donations
Sponsorship Requesting Fixed Cost
Municipality Help with Network Design
Projected Initial Total Traffic
Sponsorship to be Secured
Net from Donations
Cost of Requesting Sponsorships
Raising Donation Fixed Cost
Projected Initial Network Characteristics
Network Design Cost
Equipment and Software Cost
Projected Initial Bandwidth to Internet
Net from Sponsorships
Projected Initial Bandwidth Sharing
Municipality Help with Setup and Resources
Setup and Installation Cost
Projected Initial Traffic to Municipal Resources
Community Help with Setup
Initial Bandwidth to be Purchased from ISPs
Integration Cost
Other Infrastructure Cost

Figure 2. SD model for initial setup cost.

Using this model, total cost and total deductions are easily calculated. Percentages of each cost item with respect to total cost are calculated. The budget can be compared to the final cost. The net present values can be easily calculated with proper formulation.

Initial demand is estimated in the model which is characterized by:

1. Projected initial total traffic
2. Projected initial traffic to wireless network, the underlying wireline network and the IT infrastructure
3. Projected initial traffic to the Internet
4. Projected initial bandwidth sharing (sponsored by the sponsors): This value is provided as an estimate into the model. This value is estimated from service area characteristics in a separate SD model and fed into this model as a variable.
5. Projected initial bandwidth to be purchased from ISP

Economic development opportunity variable was not included in the characterization since it is not applicable for all service areas, but applicable to service areas with business development objectives such as business districts. It indicates how much economic development opportunity exists in the service area and is one of the variables used to determine the size and coverage of the wireless network. If there is not much opportunity, then no big investment will flow into the network and therefore no big wireless network.

Although initial setup cost estimation could be done with a simple spreadsheet model, SD modeling is still useful for visualizing different cost components and how they are related. It is also helpful to run sensitivity analysis on how different scenarios impact the cost and for estimating the cost spanning over the duration of the setup of the wireless network. Sensitivity analysis could estimate the total cost along with other variables in different scenarios with different size of the service area, amounts of community volunteering, municipality help, grants, involvement and sponsorship from businesses, and with varying levels of success in crowdfunding and crowdsourcing, and with varying cost components such as equipment, setup cost, consultancy cost. It would be easy to see if the setup cost is within the budget under what combinations of other variables. Decision makers could use sensitivity analysis to balance various variables to achieve the cost objectives.

5.2. Maintenance cost estimation

Maintenance cost includes ongoing capital expenses and ongoing operational costs. The first is due to upgrades in response to increased demand and better understanding of usage characteristics. The second one includes costs for bandwidth, electricity, contractors, equipment maintenance.

Figure 3 shows an SD model for estimating the total maintenance cost. This model has feedback loops and non-linear relations where the advantages of SD can be realized more compared to the model in **Figure 2**. The model does not show all the variables, rather it shows the different characteristics for simplicity. Variables exist within the characteristics in the model as per characterizations outlined in earlier sections.

Users in this model are classified into PotentialUsers, Users, Quitters and BlockedUsers. The first three are measured as stock variables in the SD model and are related to the adoption of the wireless network by users. There will be blocked users and that is expected by design as explained earlier. An ongoing evaluation of the performance of the wireless network using various measurement tools should help the decision makers if it is up for upgrades.

The number of potential users may change based on demographics, economic, or other factors as well as consumer behavior. Potential users adopt based on not only the intrinsic utility but also on expectation of future utility. Similarly for the churn of existing adopters. Network externalities play an important role on the utility in addition to the intrinsic utility from the wireless network itself. The utility of the service for the user depends on interconnections among the users and utility they receive from the other sides of the platform. Existing users may become quitters depending on their patience levels due to low service quality and poor customer service. Quitters may become adopters based on the come-back fraction variable which is set a value based on the expected future utility.

One important variable is area attractiveness. This depends on many factors. It is one of the factors that attracts new users or leads them to Quitters stock. It is influenced by the number of users. So, there are positive and negative loops between the attractiveness and the number of users yielding an s-shaped curve. Area attractiveness has similar relationship with

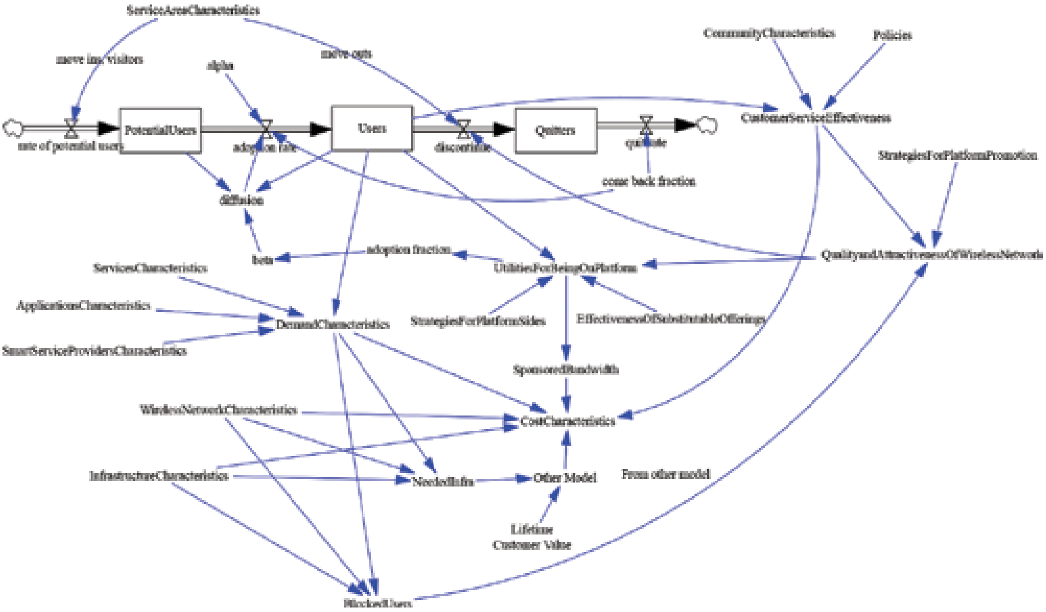


Figure 3. SD model for maintenance and operation cost.

bandwidth sponsoring, yielding again s-shaped curve. The model does not show a stock variable for area attractiveness. It is retrieved from the Service area characteristics variable. A separate SD model measures the area attractiveness as a stock variable.

Utilities for being on platform has a separate economic utility variable for each side of the platform, that is, for users, bandwidth sponsors and smart service providers. Strategies for platfom sides includes assigned values for the effectiveness of considered strategies for increasing the utilities of platform sides. Sponsored bandwidth is an aggregation of all sponsored bandwidth from any side on the platform including users, businesses and smart service providers as they all could sponsor bandwidth. For business districts, the model assumes more bandwidth sharing by the local businesses.

Ongoing Demand Characteristics for this model includes total demand by users, bandwidth per user, demand for other smart service providers, bandwidth by IoT devices and wireless sensor networks, fraction of bandwidth from smart service providers to the Internet, total bandwidth in network, total bandwidth to the Internet, bandwidth to buy from ISPs. The amount of bandwidth sharing is to be calculated over the given period, e.g. 4 years. With this model, total cost for a unit period, e.g. each month, is estimated over a duration, e.g. 4 years. SD is good for this type of calculations for the reasons mentioned earlier.

Ongoing capital expenses in the total maintenance cost are the cost of needed upgrades and expansions to the wireless network and infrastructure. This cost is calculated similarly to calculating the cost of infrastructure in the linear Initial Setup Cost model in **Figure 2**.

Ongoing operational costs in the total maintenance cost include:

1. Cost of Customer Service: this relates to customer service effectiveness, mainly the cost of part-time admins and customer service representatives.
2. Cost of Bandwidth to Buy From ISPs: this is related to the difference between the sponsored bandwidth and the total bandwidth demand to the Internet over the period.
3. Electricity Cost: this is for all APs, nodes, servers and other equipment from Wireless Network Characteristics and Infrastructure Characteristics.
4. Device Maintenance Cost: this includes replacing and repairing the APs and other equipment subject to the availability of the devices, networks and infrastructures (but not including the IoT devices and sensor networks of smart service providers).

5.3. Different scenarios for cost

The model can be run for simulating different scenarios. With simulations and sensitivity analysis, this model could be a powerful tool to answer many questions. Not just estimations of various cost components, but also relations among other variables can be analyzed. Some questions to answer using this model include:

- How many users use the system and how that number changes over time. Similarly for blocked users and for uses who quit due to dissatisfaction.

- Can the network provide QoS (enough bandwidth) for different usage patterns?
- How the municipality role impacts the success with respect to achieving the number of users and keeping the cost within the budget.
- How the rollout strategy impacts the user adoption.
- How the maintenance and management policy impacts the user adoption.
- Is the maintenance cost within the budget?
- How much grant is needed?
- How much bandwidth from sponsors is needed?
- How different strategies for incentivizing different sides of the platform affect the cost.
- How different strategies for incentivizing the sponsors work?
- How the network effects are?
- How the utilities change with different strategies, with different numbers of users and blocked users, with different levels of customer service and quality of the wireless network. How all values change in response to service area characteristics particularly its attractiveness.

Various scenarios can be analyzed by sensitivity analysis. Different triggers may be programmed for different variables at certain intervals to see how the cost fluctuates over a period. Different statistical distributions may be used over time for different cost items and for the characterization of community, service area and municipality.

6. Developing models for estimating benefits, drawbacks and risks

A smart community wireless platform offers benefits to the platform sides. Every such platform is unique and offers unique benefits to the platform sides. Different platforms would be built with different objectives by different communities. The success of the platform is evaluated with respect to the objectives of the platform.

An exercise to be done early in the planning phase is to identify the objectives of the platform. However, there are challenges in doing so: What the objectives should be? What benefits are sought? What beneficiaries are considered? Some benefits to one side may not be as beneficial to another side. There could be conflicting interests from different platform sides and therefore network effects may not be all positive. There could be even conflicting interests within the same platform side, for example residential users vs. visitors, retail businesses vs. others, commercial smart service providers (waste management) vs. municipality smart services (smart parking).

There are also challenges with measuring the benefits: Some benefits are tangible, some are not. Some are short term, some are long term. Some are direct, some are indirect. Some will influence other benefits in positive way, some will do in negative way. What are the conceived benefits? How to measure if the platform provides the conceived benefits? How to define the success of the platform and how to measure the success? What are the risks for the success and how to mitigate them? How policies and strategies are related to the success? What are the drawbacks and how to limit them? How do the drawbacks, the risks, the policies and strategies, the mitigation plans impact the success?

To address these issues, we propose a methodology for developing models to be used for estimating and measuring benefits, drawbacks and risks. These models could be used by local government officials, communities and local smart service providers in decision making. For developing models, we suggest the following methodology:

1. First characterize the objectives, benefits, drawbacks, risks, mitigation plans, policies and strategies for a given service area or a community.
2. Based on these characterizations, characterize the service area and the community. Identify the dynamics involved and variables to be used in estimation.
3. Then follow the same intelligence framework for building models.
4. Use the models for simulations and sensitivity analysis in early decision making.
5. As more insights are obtained and more data is collected, build statistical, economical and machine learning models, integrate them with the SD models, validate and refine the SD models.
6. Continue simulations and sensitivity analysis using the refined models as they are valuable tools in estimating and deciding if the platform would be feasible for a longer term, and to apply which policies, strategies and mitigation plans for the success.

We start applying the methodology by characterizing the dynamics in subsections below.

6.1. Characterization of objectives

The common objectives for smart community wireless platforms are (1) offering public safety and city services in the community and civic engagement, (2) closing the digital divide, (3) convenient services for citizens and users by enhancing digital experience, (4) economic development [7]. The characteristics about the objectives should determine what objective(s) are most relevant and with what relative weights for the community.

6.2. Characterization of benefits

The platform offers benefits to different sides as we summarize below. The exercise for characterizing the benefits should provide values for the listed benefits below.

6.2.1. Benefits to users

Access to free wireless Internet for citizens: Provides citizens with Wi-Fi experience and location-based services. Citizens can access the Internet over their smartphone, tablet, and other computing devices when they are in public spaces and on the move. They have access to city information and city services anytime. A community app will help improve the digital experience of the citizens. They have access to smart services provided by the smart service providers over the platform. The number of smart services matters. The more the number of smart services, users are more likely to have higher utility with the platform. However, this number by itself is not sufficient to increase the utility of the users, the quality as well as the functionalities provided by the smart services are significant factors.

Access to free wireless Internet for visitors: Enhance the visitor's experience. Community app for visitors will enhance their visiting experience, for example, from parking to shopping and attractions.

6.2.2. Benefits to bandwidth sponsoring businesses

Will the platform lead to economic growth such as more businesses, revenues, jobs, transactions, wages? Businesses can do better targeted marketing thanks to analytics which would yield density/utilization at given time of day or day of week, people flows/footfall, time spent in the area, first time versus repeat visitors. Location-based services offer new insights about user behavior that can also be leveraged by local businesses/retailers to do better targeted offers. Businesses can operate and adjust (hours, number of employees) based on the location data. Businesses could take advantage of real-time analytics for prediction of repeat visitors as well as new visitors based on similarity, and can customize their marketing strategies. Shopping centers can boost footfall by enabling shoppers to stay connected to social networks and share their experiences. New startups may appear for example for offering smart services over the platform. Innovation is fostered through university collaborations and entrepreneurial engagement over the platform.

If businesses and organizations benefit from the platform, these same businesses may also give back. Larger businesses may provide bandwidth and others may act as root access sites and/or connection points. With such returns, positive network effects are realized.

6.2.3. Benefits to smart service providers

Smart service providers may benefit by connecting their wireless sensor networks and IoT devices with the wireless network. They can offer smart services for the community. Municipality is also a smart service provider and may place their IoT devices/networks like the commercial ones for smart environment monitoring, smart light and street management, weather monitoring, traffic monitoring, smart public safety. Benefits include reduced cost by using wireless network for delivering smart services and reducing cost of operations with smart technology. For example, a smart service provider would save by having access to almost real-time data of its devices. More benefits are realized when these platforms integrate to a larger smart city wireless platform [5].

6.2.4. Benefits to community and city

One conceived benefit is a safer community with police, fire, emergency medical response teams using the wireless network for safer streets and neighborhoods. This helps with public safety, incident response, law enforcement, and keeping stores safe from thieves.

Efficiency is expected to improve in delivering public services with municipality using the wireless network for their services and citizens accessing those services. This results in lower energy and maintenance costs as well as more revenue from city services for example with paid parking. The platform helps the city with better leveraging of existing assets, better traffic management, improved planning, better ROI and greater savings for city reinvestment. It helps the community economically by boosting economic growth and city prospects, and it may help with innovation. It helps with improved education for closing the digital divide and greater citizen compliance. It helps the communities to be greener and elegant with smart waste and trash management.

About the benefits listed above, some are measurable such as smart lighting by comparing the electricity cost. Others are harder to measure such as benefits of reducing digital divide, and are over long term. Some are direct for example bringing in more visitors. Some are indirect such as crime rate reduction and increase in quality education which are not just due to the existence of the wireless platform but other factors as well.

6.3. Characterization of drawbacks

The platform has some drawbacks. There are inherent difficulties with building the platform as well as operating it. The characterization of drawbacks should provide values for them which are summarize below.

6.3.1. Harm to private investment

One main positive outcome with this approach is that it does not lead to legality issues. This is because the platform is owned and sponsored by the community. In other approaches where the municipality owns the network, ISPs can sue cities for creating municipal wide Wi-Fi networks because cities would be competing with free market practices. In general, the city cannot monetarily benefit from a municipal wireless service in many states. Even with this model, the city should be careful in their policies and strategies on supporting the community platforms.

A risk with this model is it may distort the competitive markets and private investors. As a result, service providers feel discouraged from entering the market. Another risk is if the private investment may not update the cellular infrastructure in the community, but rather prioritize other communities that do not have such platform. On the contrary, this platform could lead the competing private commercial providers to enhance their services and offer higher quality services than the community wireless network and/or reduce their fees. The community wireless networks will produce more network traffic to/from the Internet and depend on ISPs enhancing their broadband services. When such community platforms are integrated to form a city-wide platform, the bigger platform can be used as a tool for bringing in more and diverse private investment. This is a subject we elaborate in [5].

6.3.2. Wireless network quality and availability issues

The wireless network and the infrastructures in **Figure 1** may not be as high quality as commercial offerings. How about the network problems, for example, access point or network element problems? Who will solve them? How about any customer service? Would the network be reliable and available and stable to run smart services? The availability of the network and the infrastructure may be less than adequate to run smart services. Similarly, the QoS supported may not be comparable to commercial wireless services.

Nonetheless, the platform should be positioned to support only non-critical services. Since this is a community wireless network, it does not necessarily cater for high performing and critical networked applications unless they are needed by the community and the community takes charge of supporting them.

As discussed in System Architecture section, the platform offers limited bandwidth for users and enforces a cap in total upload/download per period (e.g. day). Also, it enforces a limited number of connected users at any time. One risk is if it cannot attract enough sponsors over time to increase the limits. Another risk is the shortage of private investment to enhance the local broadband infrastructure and thereby the community network cannot have enough bandwidth to/from the Internet.

6.3.3. Security issues

Various security issues exist for this platform. As an example, hackers can create rogue wireless networks to lure the users into their network as opposed to the community wireless network. The platform should provide security guidelines for the users, and should monitor emergence of such rogue networks.

6.3.4. Drawbacks to the community

The sought benefits of the platform for the community may not be realized. Rather to the contrary, some drawbacks may appear. As an example, we can mention disruption to conventional way of life and businesses. This is due to embracing an all-digital life and isolating ourselves from traditional human interactions. Another drawback is increasing the digital divide. This happens when the platform becomes a playground for tech savvy but leave behind the other interest groups which are not very tech savvy [4].

6.3.5. Drawbacks to users

One drawback to the users with this platform is the privacy. Users give up on their privacy in return to getting free wireless service. Other drawbacks include limited bandwidth, low quality and security issues in the wireless network during use. Another drawback is the limit in total supported users at a time, which could block some users. Offering less than adequate customer service is another drawback.

6.3.6. Management issues

This platform relies on joint efforts from the municipality and the community. What happens when the community disagrees on the objectives and policies. There could be conflicting interests within the community about the objectives of the wireless platform. Policies are hard to agree upon and implement. Voluntary nature of most tasks may lead to slow progress on the development and inefficiency on the operations.

6.3.7. Financial issues

This platform relies on various funding sources. If funding cannot be granted, the platform may degrade and may not serve its objectives and may be abandoned by the sides, and eventually by the sponsors. This would result in a failed project and wasted resources.

6.4. Characterization of risks

The community decides on what level of risks to accept and what mitigation plans to consider and strategize. Most drawbacks outlined in the Characterization of Drawbacks section are risks that need to be managed. We can categorize types of risks as follows:

1. Risks of hitting the drawbacks above
2. Risks with the development of the platform
    a. Risk not getting enough help from the municipality
    b. Risk of political change in the city, disagreements between the municipality and the community
    c. Risk of project falling apart due to management and policy reasons, conflicting interests within community
    d. Slow progress due to bureaucracy and volunteering
3. Risks during operations
    a. Quality issues not handled
    b. Sides losing interest
    c. Not enough benefits realized
    d. Community not becoming part of smart city wireless platform [5].
4. Risk of pushing out private investment

The exercise for characterizing the risk should provide values for the levels of risks.

One risk is when the community does not get enough help from the municipality as the platform relies on municipality for various help including help with funding resources. Another risk is if the perception changes with political change by a new city administration and if the new administration does not consider the community platforms a priority. This will leave communities without help from the municipality. Another risk is that the progress may be slowed by political maneuvering and complex coordination processes. Another risk is conflicting views between municipality and the community. This relates to defining roles and the operational policy for preventing conflict between community and municipality. Conflict may lead to a risk of community not being part of the bigger smart city wireless platform or leaving it [5].

One risk is when the community cannot secure enough funding for building a platform even at a small scale.

One risk is when stakeholders, particularly bandwidth sharing entities, may not feel incentivized to bandwidth sharing. If the community does not have access to shared bandwidth, the cost of the network increases dramatically. Businesses such as hotels, shopping malls/centers have their own wireless networks and loyalty programs. Why should they be part of the platform? The benefits should be explained well to the businesses. This depends on effectiveness of community for convincing the partners. There is risk with sustainability due to lack of sufficient funds for maintenance and upgrades, and low adoption by intended beneficiaries.

There is risk of cellular providers not improving the service or not updating to latest technology if the mesh networks become widespread in the city. Another risk is wireless hotspot providers may end their services due to community wireless network. To mitigate these risks, the community should not build the network to compete against commercial providers as outlined in Strategies For Platform Promotion and Positioning section.

Another risk is with causing divides in the community, for example between geeks and non-geeks when tech savvy members controlling the platforms and others are falling behind. As a mitigation, the community should seek inclusion of all, not just the tech savvy.

6.5. Characterization of success

The success of the platform is evaluated with respect to its objectives. One success factor is how the economic utilities of the platform sides increase. Others include:

- How policies and strategies are contributing positively toward the objectives
- How the benefits are realized, how the cost is kept to minimum with grants and crowdunding
- Whether the objectives were really the right ones, or unachievable or unrealistic or unbeneficial objectives were pursued

One success indicator is the ratio of sponsored bandwidth to total bandwidth, which is an indicator to effectiveness of the community in getting bandwidth sponsors on the platform. Performance related success criteria include QoS, reliability and availability measures of the network, time to respond to tickets, whether the network can support maximum number of non-blocked users with predictable QoS. The exercise for characterizing the success should provide values for the above.

6.6. Service area and community characteristics

Community characterization is done by characterizing the above dynamics for each service area in the community that is by identifying the objectives, sought benefits, risks, mitigation plans, drawbacks, policies and strategies, success criteria for each service area. For this characterization, the size of the service area matters. The resources such as social and non-profit organizations and businesses in the area matter. Opportunities such as economic development opportunities in the service area matter. How the municipality sees the service area matters with respect to whether municipality considers significant investment or not in the area, and what social initiatives and public services are planned. Existence of substitutable offerings matters.

7. SD model estimating benefits, drawbacks and risks

Once the characterization phase is complete, the methodology suggests following the same intelligence framework for building models to estimate benefits, drawbacks and risks. In this paper, we develop a generic SD model taking into consideration the dynamics we characterized in earlier sections. The generic model in **Figure 4** shows how these dynamics influence each other. Then we will explain how the generic model could be instantiated for specific service areas.

The generic model shows the impact of success on users and the impact of user adoption to observed benefits. More users could lead to more benefits initially, but since some users will be blocked after the maximum allowed users, the quality and attractiveness of the wireless network will degrade. That will cause some users to quit, and will impact the observed benefits negatively. Positive loop from economic utility to observed benefits exists: when the quality and attractiveness of the wireless network is high, the economic utility for being on the platform will be high for users and other platform sides. When more users join, the observed benefits will increase to some extent. Negative loop also exists from low success to benefits: the low success on the objectives will degrade the quality and attractiveness of the wireless network, which will lead to lower economic utility for users and sponsors on the platform, which will reduce the amount of observed benefits. With small number of users, no big benefits could be achieved. If the critical mass is not reached, it may be partly due to non-effective policies and strategies.

Most benefits are intangible. In the simulation, some weights are assigned for benefit related attributes. Another advantage with SD analysis is to be able to simulate the uncertain benefits of the platform and compare the benefits under different scenarios through sensitivity analysis. Another challenge with measuring benefits is that some benefits are realized over a long term. Different statistical distributions may be used to take effect over time for the characterization of

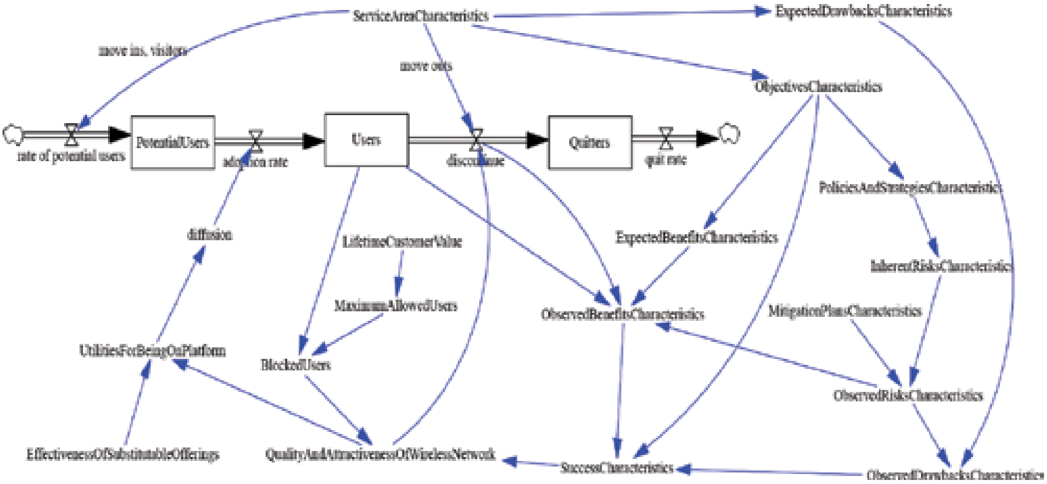

ServiceAreaCharacteristics
ExpectedDrawbacksCharacteristics
move ins, visitors
move outs
ObjectivesCharacteristics
rate of potential users
PotentialUsers
adoption rate
Users
discontinue
Quitters
quit rate
PoliciesAndStrategiesCharacteristics
diffusion
LifetimeCustomerValue
ExpectedBenefitsCharacteristics
InherentRisksCharacteristics
MaximumAllowedUsers
ObservedBenefitsCharacteristics
MitigationPlansCharacteristics
UtilitiesForBeingOnPlatform
BlockedUsers
ObservedRisksCharacteristics
EffectivenessOfSubstitutableOfferings
QualityAndAttractivenessOfWirelessNetwork
SuccessCharacteristics
ObservedDrawbacksCharacteristics

Figure 4. Generic model for measuring benefits of smart community wireless platform.

community, service area and municipality, and that way, their impacts on the benefits are measured over a long term. Triggers for various dynamics at certain times help with analyzing the impacts over a long term.

The model contains all different characteristics and their relationships. Those characteristics include various variables corresponding to different aspects of characterization we outlined in earlier sections, although the variables are not visible in the diagram. When the model is instantiated for a specific service area, only the relevant variables are populated for each characteristics. Otherwise, a model that contains most variables is hard to build, nor would be helpful because of huge diversity in objectives, benefits, drawbacks, risks, policies and strategies. Rather the generic model is instantiated for each service area to simulate the different dynamics pertinent to the service area.

A typical instantiation of this generic model involves one-to-three objectives/benefits, one drawback, one risk and one mitigation plan in the model. Also, typically one smart service is incorporated into the model to simulate how it would impact the utility of the platform sides. More specifically, the smart service could be local to the service area and would not be available via the Internet in the instantiated model. This generic model allows analyzing the network effects of a single smart service. An example is how a new smart service by itself influences the network effects and the utilities in the service area. Another example is how community beacons or augmented reality introduced in a business district influence the network effects and utilities.

An instantiated model is used for estimating the benefits and drawbacks, and for analyzing the relations among benefits, drawbacks, risks and mitigation plans in existence of network externalities for a service area. When enough data is not available, SD modeling and simulation is still helpful by performing sensitivity analysis based on assumptions, heuristics, hypothesis, expert opinions, estimations and observations for providing insights to many managerial questions.

As the next steps in the methodology described earlier, the model is further verified and validated as more data becomes available. With more data about the dynamics being available while the wireless network is operational, models that use statistics and machine learning are built for clustering, classifying and predictions. Factor analysis is done on what dynamics affect the success of the platform the most. Statistical methods are used to see significant difference between different platforms and between different scenarios, and to test hypothesis about the relations among different dynamics related to the platform. Machine learning methods are built to analyze collected usage data per community network for predicting future use and demand forecasting, finding out covariance matrix, significant parameters, association rules regarding the success of the platforms. All these statistical, machine learning and economical models are integrated with the SD models and the SD model is validated and tuned using available data. With data, it is possible to do comparison between different service areas. We believe statistical, economic and machine learning methods alone are not sufficient to analyze complex platforms such as this one, and SD is most suitable to be used together with these methods, hence a more powerful intelligence framework for better analysis and understanding can be constructed [11].

The model should run over time for a service area to see how the benefits and drawbacks are realized over a period. The system should analyze all possible improvement strategies and tradeoffs, balancing required budgets and expected benefits.

8. Smart City wireless platform

When different smart community wireless platforms come together, is it possible to create a bigger platform for the whole city? We call this new platform the smart city wireless platform. In the bigger city-wide deployment of a smart city wireless platform, the municipality would be the sponsor of the platform to ensure order and control mechanisms. This platform requires additional infrastructure elements in the system architecture and has more sides compared to the smart community wireless platform. We explore it in [5].

9. Conclusion

A wireless network (e.g. a mesh Wi-Fi network) covering most of the city is a significant contributor toward being a smart city. Such a network offers many benefits but there are technical, economical and policy challenges for building and operating one. This paper presents a model where municipality, communities and smart utility providers work together to create a platform, the smart community wireless platform, where different sides work together toward achieving smart community objectives. The main advantage with this platform is that communities have clear objectives and needs, and have better predictions about the demand, and are small and manageable in sizes. The municipality does not allocate big budget for initial and ongoing cost. The network provides bandwidth for smart IoT devices and access to the services offered by smart service providers. This model allows collaboration among communities, municipality and smart service providers.

One question is how much would be the total cost of building and operating such a platform. To estimate the cost, relevant dynamics should be identified and characterized. An intelligence framework that incorporates SD modeling with statistical, economical and machine learning methods is very useful for estimating the total cost of smart community wireless platforms under various conditions and scenarios. In this paper, we developed models for estimating the initial and maintenance costs, and outlined how these models could be used to analyze different dynamics and scenarios. These models can be used by the community which is the platform sponsor and by the city which is a main supporter of the platform. Through simulations and sensitivity analysis, these models could provide insights about different cost components as well as about other dynamics of the platform.

Another question is how to measure the benefits and drawbacks of these platforms to estimate the returns on investment over a period. Another question is how to analyze the risks and mitigation plans for the success. To measure the benefits, relevant dynamics should be identified and characterized. The characterization phase should consider objectives, benefits, drawbacks, risks, policies, strategies and criteria of success for a specific service area in a community. The same intelligence framework that we used for estimating the total cost is applicable for estimating the benefits under various conditions and scenarios. In this paper, we developed a generic SD model for estimating the benefits and drawbacks, and for incorporating the causal loops among benefits, drawbacks, risks and mitigation plans in existence of network externalities. We outlined how the generic model could be instantiated for specific dynamics and to analyze different scenarios.

Another question is how the city could inspire and assist the communities to build their community wireless network, and then coalesce them for a city-wide wireless network. We address this question in [5].

Our future work includes running the models for different instantiations and comparing results. Our future work also includes combining the cost and benefit models together and running them to measure returns on investment for different scenarios.

Author details

Sakir Yucel

Address all correspondence to: yucel@bluehen.udel.edu

University of Delaware, Newark, Delaware, USA

References

[1] Abdelaal A. Social and Economic Effects of Community Wireless Networks and Infrastructures. IGI Global; Feb 28, 2013

[2] Shin S, Tucci, Lesson JE. Lessons from WiFi municipal wireless network. AMCIS 2009 Proceedings; 2009. http://aisel.aisnet.org/amcis2009/145

[3] Evenepoel S, Van Ooteghem J, Lannoo B, Verbrugge S, Colle D, Pickavet M. Municipal WiFi deployment and crowd-sourced strategies. Journal of the institute of Telecommunications Professionals. 2013;**7**(1):24-30

[4] Byrum G. What are community wireless networks For? The Journal of Community Informatics. 2015;**11**(3)

[5] Yucel S. Smart city wireless platforms for smart cities. The 14th International Conference on Modeling, Simulation and Visualization Methods (MSV'17), July 17-20, 2017. Las Vegas, Nevada, USA

[6] Lee SM, Kim G, Kim J. Comparative feasibility analysis of Wi-Fi in metropolitan and small municipalities: A system dynamics approach. International Journal of Mobile Communications. 2009;**7**(4):395-414

[7] Kim G, Lee SM, Kim J, Park S. Assessing municipal wireless network projects: The case of Wi-Fi Philadelphia. Electronic Government, An International Journal. 2008;**5**(3):227-246

[8] Eisenmann T, Parker G, Van Alstyne M, Platform Networks – Core Concepts, Executive Summary, http://ebusiness.mit.edu/research/papers/232_VanAlstyne_NW_as_Platform.pdf

[9] Vural S, Wei D, Moessner K. Survey of experimental evaluation studies for wireless mesh network deployments in urban areas towards ubiquitous internet. IEEE Communication Surveys and Tutorials. 2013;**15**(1, First Quarter)

[10] Abujoda A, Sathiaseelan A, Rizk A, Papadimitriou P. Software-defined crowd-shared wireless mesh networks. Computer Networks. 24 December 2015;**93**(Part 2):359-372

[11] Yucel S. Delivery of digital services with network effects over hybrid cloud. The 12th International Conference on Grid, Cloud, and Cluster Computing, GCC'16: July 25-28, 2016, Las Vegas, US

[12] Yucel S. Evaluating different alternatives for delivery of digital services. The 12th International Conference on Grid, Cloud, and Cluster Computing, GCC'16: July 25-28, 2016, Las Vegas, USA

[13] Yucel S, Yucel I. Estimating the cost of digital service delivery over clouds. The 2016 International Symposium on Parallel and Distributed Computing and Computational Science (CSCI-ISPD), Dec 15-17, 2016, Las Vegas, USA

[14] Yucel S, Yucel I. A model for commodity hedging strategies. The 13th International Conference on Modeling, Simulation and Visualization Methods (MSV'16), July 25-28, 2016, Las Vegas, USA

[15] Schuurman D, Baccarne B, De Marez L, Mechant P. Smart ideas for smart cities: Investigating crowdsourcing for generating and selecting ideas for ICT innovation in a city context. Journal of Theoretical and Applied Electronic Commerce Research. Dec 2012;**7**(3), ISSN 0718-1876 Electronic Version

Data Simulation and Trend Removal Optimization Applied to Electrochemical Noise

Victor Martinez-Luaces and Mauricio Ohanian

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/intechopen.73831

Abstract

A well-known technique, electrochemical noise analysis (ENA), measures the potential fluctuations produced by kinetic variations along the electrochemical corrosion process. This practice requires the application of diverse signal processing methods. Therefore, in order to propose and evaluate new methodologies, it is absolutely necessary to simulate signals by computer data generation using different algorithms. In the first approach, data were simulated by superimposing Gaussian noise to nontrivial trend lines. Then, several methods were assessed by using this set of computer-simulated data. These results indicate that a new methodology based on medians of moving intervals and cubic splines interpolation show the best performance. Nevertheless, relative errors are acceptable for the trend but not for noise. In the second approach, we used artificial intelligence for trend removal, combining an interval signal processing with backpropagation neural networks. Finally, a non-Gaussian noise function that simulates non-stationary pits was proposed and all detrending methods were re-evaluated, resulting that when increasing difference between trend and noise, the accuracy of the artificial neural networks (ANNs) was reduced. In addition, when polynomial fitting, moving average removal (MAR) and moving median removal (MMR) were evaluated, MMR yielded best results, though it is not a definitive solution.

Keywords: electrochemical noise, data simulation, signal processing, trend removal methods, noise filtering, artificial neural networks

1. Introduction

To accurately determine the life expectancy of an industrial component, it is necessary to quantify the metal deterioration by environmental influence —for example, construction, automotive, naval and aeronautic industries, among others. For this reason, it is important to obtain accurate methods to measure and predict deterioration processes [1].

A well-known methodology is ENA, which measures variations in current and/or potential fluctuations, produced by fluctuations in electrochemical process kinetics. An important advantage of this corrosion measure technique is that no external signal perturbations are necessary, as the electrochemical system uses natural techniques for measurement. Additionally, measurement devices are better in terms of affordability in comparison with other techniques [2]. Moreover, ENA provides information in terms of current and potential fluctuations in low amplitude and frequency and presents more complexity in experimental data treatment. The application of ENA technique yields information about the corrosion process mechanisms, kinetics, and morphology [3] from the calculation of the parameters of noise resistance ($R_\mathrm{N} = \sigma_V/\sigma_I$), localization index ($LI = \sigma_I/I_{RMS}$), and power spectral density (PSD).

In this process, an electrochemical signal that follows a particular trend is perturbed with a certain noise. In real conditions, noise appears as a result of corrosion reactions, while the trend is due to hydration processes, species dissolution, and thermal cycling, among other processes, which are not necessarily associated to the material deterioration. Therefore, to understand corrosion behavior, it is crucial to detrend the signal function isolating this noise. This procedure is called trend removal [3]. Several methodologies have been applied to implement it, especially statistical procedures [4]. Despite having been proved to be useful in some cases, a really accurate procedure for trend removal is still an open problem.

In fact, it is relatively easy to remove noise from a noisy function by polynomial approximation (ANNs) [5] or genetic programming [6, 7]. However, it is extremely difficult to do the inverse filtering. In fact, as shown in Section 3, trend removal cannot be performed accurately by traditional techniques such as polynomial approximation. If this technique is used, low-level relative errors in trend approximation become unacceptable to high-level ones in case of noise approximation, since errors and noise usually have the same order of magnitude. For this reason, other algorithms and computational techniques—as ANNs, analyzed in Section 4—must be studied in order to improve the traditional detrending methods, proposing new feasible and adaptable trend removal tools for different purposes.

Additionally, it is important to mention that the main disadvantage of ENA technique is the high dispersion observed on experimental results [2]. In a previous work [8], three different factors (electrolyte, frequency simulation, and trend removal) were analyzed with a view to determine causes for the observed high dispersion. The experiments done, modifying these parameters, showed that the trend removal method is the one which best explains this phenomenon.

In order to assess the performance of the different methods, it is a must to know exactly both trend and noise for a particular signal. Then, the error levels obtained using these methods can be statistically compared.

Due to these reasons, trend and noise were both computer simulated, as described in detail in the next section.

2. Data simulation for ENA

2.1. Analytical trend simulation

In order to simulate the artificial trend, nontrivial functions must be used in the process. As mathematical methods can easily be used to approximate them, trigonometric functions, polynomials, and exponentials can be considered as trivial trend lines. This is a relevant aspect that should be taken into account when trend removal methodologies are evaluated.

For this purpose, we used two different curves as trend lines, since they are nontrivial and at the same time their graphics look like the experimental curves obtained in previous research.

The first trend line considered was: $\frac{f(t)=ac^{t/3}}{\Gamma\left(\frac{t}{3}+1\right)-b}$ (Curve 1), where $\Gamma(x) = \int\limits_{0}^{+\infty} t^{x-1}e^{-t}dt$ is the Euler's Gamma function. This transcendent function cannot be approximated by elementary methods, and at the same time, its shape is similar to one of our experimental curves, and for this reason it was selected among others studied in a first approach.

A second trend line chosen for evaluating the detrending methods was the Lorentz's function: $f(t) = y_0 + \frac{2A}{\pi}\frac{w}{4(t-x_c)^2+w^2}$ (Curve 2), which also looks similar to one of the experimental curves obtained in previous experiments. Besides, due to the Runge's phenomenon (see the similarity of $f(t)$ with Runge's function), this curve cannot be easily approximated since going to higher degrees polynomial interpolation does not improve accuracy.

2.2. Noise distribution

As in case of trend, the goal with noise is to simulate real conditions as much as possible. In real conditions, noise is unpredictable, and so a random function for noise data generation should be used.

In a first approach, noise data generation was implemented using an inverse Gaussian distribution. To do this, a numeric algorithm was applied [9] by splitting into three sections the Gaussian function according to the derivative level (low, medium, and high) and then applying different algorithms to each section for error minimization. Once done, noise generated function is added to trend function, obtaining a noisy signal adequate for trend removal testing.

3. A first approach about detrending methods

Some theoretical aspects of the trend removal problem were developed by K. Hung Chan and colleagues [10]. The authors presented differences between the most used trend removal methods: first difference method (also named point-to-point method) and least squares. They utilized a linear trend superimposed with white noise and the first difference method gave power spectral density (PSD) exaggerated at high frequencies and attenuated at low frequencies. On the contrary, the regression residuals method generates results with PSD exaggerated at low frequencies and attenuated in the high-frequency region.

Mansfeld et al. [11] studied the characteristics of detrending in ENA data with an experimental and theoretical approach. Thus, by applying linear fit to the trend of experimental data, they obtained a good concordance between the noise spectra after detrending and impedance spectra. The authors found that MAR, previously proposed by Tan et al. [12], caused erroneous ENA results.

The performance of most used trend removal methods (analogical high-pass (HP) filtering, digital high-pass filtering, MAR, and polynomial fitting) was analyzed by Bertocci et al. [13]. They worked on a simulated data set achieved by superimposing white noise to a linear trend and applied the detrending methods. They studied the box size influence on MAR, concluding that a small box size effectively removes the trend without phase shift. However, MAR also removes the signal information at low frequency. A high-order MAR (greater box size) generates alterations in the signal shape. With the intention of evaluating the produced attenuation, the authors used polynomial detrends of different orders. The components in frequency $1/T$ and $2/T$ window (T being the experimental period) were eliminated when a fifth-order polynomial was used. The best results were obtained when the polynomial methodology was shared with prepossessing windowing. By using the Chebyshev filter (digital processing) of different cutoff frequencies, the authors recovered the white-noise PSD. They concluded that the use of high-pass (HP) analogical filtering is the ideal method to eliminate low-frequency components from the experimental signal. Although filters of cutoff frequency near to $1/T$ with low intrinsic noise were too costly. Besides, the major drawback of analogical filters in real-time processing is related to the oscillations that the components have in abrupt signal variations, that is, when switching on the system.

Ohanian et al. [8] found a great dispersion in the experimental ENA results on low-alloy steel performed in saline solution. The authors attributed the excessive dispersion mainly to the detrending method used. On a simulated data set, they analyzed the polynomial order influence and the aliasing phenomena.

As we mentioned earlier, there are many methodologies for trend removal to isolate ENA. In a first approach, several methodologies were analyzed, since they were studied by well-known authors like Mansfeld [11], Tan [12], and especially Bertocci and Huet [13], who evaluated the most commonly used trend removal methods. Four of the detrending methods from this group are described in detail, as they showed better performance than others that have been assessed by using the computer-simulated data. Three of them can be considered traditional methods, since they appear regularly in papers on studies performed on trend removal technique using ENA. The fourth method was proposed by our research team in order to obtain more accuracy between the simulated noise and the results of the detrending process.

The selected methods are the following:

- Polynomial fitting

This is a well-known technique widely used [14]. Trend is fitted using squares regression, and then noise is obtained as the difference between experimental and predicted data (by the regression model). In this case, the best results were obtained by using a commercial calculus package (Origin Pro 7 ®). Due to the limitations of the program, a ninth-order polynomial was utilized with the maximum order available.

- MAR-10

In this method, noise is computed using a moving average [15]. More precisely, if m_n represents the following moving average: $m_\mathrm{n} = \frac{1}{21}\sum_{p=-10}^{10} x_{\mathrm{n}+\mathrm{p}}$, then noise is computed as the difference $x_\mathrm{n} - m_\mathrm{n}$ where x_n denotes the nth simulated value. Here, the size of the box (i.e., the one with 21 points) is the result of an optimization, analyzing different box sizes currently used in previous works.

- Butterworth filtering

This method uses analogical filters [8], and it is part of the MATLAB Signal Processing Toolbox software (Matlab® Signal Processing Toolbox©). The Butterworth filter is a type of signal processing filter designed to have as flat a frequency response as possible in the passband. The filter returns the transfer function coefficients of an nth-order low-pass digital Butterworth filter with specified cutoff frequency Wn. This methodology is a commonly used trend removal method, also analyzed by Bertocci and Huet [13], among others.

- MICS

This method, named MICS (Median of Intervals and Cubic Splines) [16], is a trend removal technique that we proposed and extensively tested in a previous work [8]. The methodology is based on dividing the data set into intervals, computing the median position and the average time. The resulting points are considered as the nodes for determining a cubic spline. The noise is obtained by calculating the difference between the simulated data and the cubic spline interpolation.

In this first approach, MICS showed the best training performance. Nevertheless, due to low relative errors in case of trend that become high in case of noise because of their different magnitudes, it should be observed that in most of these methods, the relative error level is acceptable for trend but usually unacceptable and useless in case of noise.

Therefore, a different approach was needed and our next attempt was directed to the use of artificial neural networks (ANNs) for detrending.

4. Using neural networks

From the very beginning, ANNs were mostly employed on function approximation [17, 18]. From this viewpoint, trend removal can be seen as a particular case of function approximation. ANNs are also widely used as a statistical analysis tool.

The main idea of our second approach was to apply an ANN methodology to improve the trend removal procedure. Moreover, our domain (environmental corrosion) is essentially dynamic, due to high variance of functional trends, so it seems to be especially adequate the use of ANNs because of their inherent dynamic nature. Due to a big quantity of both patterns and output data, we preferred feed-forward ANNs that have proved to be especially useful in function approximation. According to Universal Approximation Theorem [19], feed-forward ANNs of three or more layers with backpropagation algorithm can approximate every continuous function (such as functions of Section 2.1), so it is particularly recommended for use in our case.

Anyway, we implemented these functions into the tool in order to make possible the "self-training" of the ANN with the generated data. Indeed, the effort is well worth the investment because this approach has the advantage of allowing the self-training along with more exhaustive testing.

With simulated data, backpropagation ANNs were trained splitting the signal in intervals and then training three different neural networks, depending on the mean derivative of each segment. Once done, a data testing set was generated for cross-validation.

The implemented backpropagation ANN is a supervised-learning neural network, which means that real results are provided with the training data set. Then, the ANN applies an algorithm based on the descendent gradient in order to minimize the mean square error (MSE)—the classical error measure in these cases. In the input layer, we began with 50 data (i.e., 50 neurons, one for each input data) although in this case we obtained poor convergence so we later reviewed this parameter, deciding to increase resolution several times until we fixed it in 400 input neurons which proved to be the best.

About the topology used, according to Funahashi's Universal Approximation Theorem, one hidden layer (three-layered ANN) is enough to approximate the noise function [20, 21]. Anyway, for performance reasons, we added a second hidden layer in the tuning phase—a practice recommended in especially complex problems [22]. In order to accelerate convergence and avoid oscillations, learning rate was fixed at 0.1 and a threshold was implemented in the backpropagation ANN. A momentum factor was also added and fixed at 0.3 [23].

Considering that we want to utilize this method to approximate different functions, the whole range was not used as an input vector [24]. This aspect is critical, since the ANN may memorize the input function, and the methodology would not be useful when working with other trend lines. For this reason, an interval pre-processing was applied, consisting of a moving window, frequently used in other cases like temporary series. We used a 40-data interval, where the original vector was split into 10 vectors of 40 data each. These 40-data vectors were then used as inputs for the ANN. **Figure 1** shows a graphical representation of the procedure decrypted.

By this approach, the ANN improved. Many optimization techniques had been suggested for feed-forward, one of them was the use of an All-Class-One-Network (ACON) structure that is to say, using many sub-nets depending on data entries. In our case, we used three sub-nets of equal topology but with different weight matrix. In the input layer, we loaded 40-data vectors in one of the three sub-nets depending on the media signal derivative on the interval (we

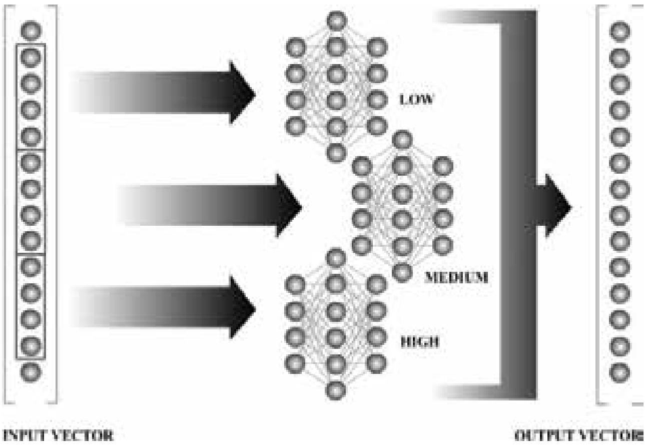


Figure 1. Split and join of signal segments.

divided the function range in three sectors, namely low, medium, and high derivative using finite differences). At the end, the obtained vectors in the output layer are joined into one vector of 400 data elements, comprising the final solution (see **Figure 1**). Different values of learning rate and momentum were evaluated during the training phase. As expected, oscillations were observed for learning rates higher than 0.3. The final value was fixed at 0.15, resulting in little oscillations when the ANN advances uniformly to the solution. Moreover, with the momentum factor added, close to a local minimum there are more steps—and also little ones—and the convergence accelerates when a local minimum is still far.

Then, the ANN was integrated with the data generators to complete the tool. The data inputs and outputs were implemented with text files or spreadsheets [24].

4.1. Training process

At first, in the training process, we used a single expression for the noise, training the ANN continuously with it. In this first phase, we could prove the ANN memorizing ability as a first step. Obviously, with single noise data, the ANN could approximate to any desired error level. In the second phase, we trained the net with a Lorentz function with fixed parameters but with random noise (inverse Gaussian distributed). We loaded the ANN with different noise values at each step, according to real conditions. In this phase, data files were saved after 10, 100, 1000, 5000, 10,000 and 30,000 iterations. These results showed a certain oscillation level but with a clear media and standard deviation-descendent direction of the mean squared error (MSE). Based on the second phase, it can be observed that with 30,000 training epochs, a 0.2% trend relative error and a 7% noise relative error are quite good when considering its random function and unpredictability. In a third and final phase, we randomized both noise and Lorentz function parameters into a determined interval according to real conditions [24]. Thus, with the most exhaustive testing, we found that relative errors were increased—as expected—being still acceptable in comparison with other methods used before.

4.2. Validation

Several statistical measures are useful for ANN validation: media and standard deviation are primary metrics. More information can be obtained with kurtosis and PSD. Also, cross-validation [19–25] was not used with the classical procedure of splitting data in training and testing sets but generating a testing set once the training phase concluded, as it was always possible to obtain fresh data. An online cross-validation was also applied, as we have already done in previous ANNs works [26], although using new testing data each step. With the optimizations applied, the convergence was good, although it could be improved with second-order [27] or even genetic algorithms [6–28].

4.3. Results obtained with ANN detrending

Once the implementation of the ANN and the generators was done, we applied the self-training of the tool with the generated data. Since Lorentz function is randomly parameterized and later perturbed with random noise (inverse Gaussian distributed), a certain oscillation level takes place, although it tends to stabilize after the 10,000 epochs. **Figure 2** shows the MSE value during the training process.

Figure 2 shows media and standard deviation of MSE values (in groups of 10) ,are shown in **Figure 3**.

According to the prediction power of the ANN, this methodology succeeds in isolating noise with an acceptable relative error level (see **Figure 4**).

As it can be observed, in **Figure 4**, the ANN had the ability to identify frequency variances and so distinguish noise from trend function. Nevertheless, the results must be checked with PSD, a suitable methodology for assessing trend removal methods.

The PSD at the end of training process, corresponding to 30,000 iterations, illustrated in **Figure 5**.

In order to prioritize inference over memorization, a suitable alternative is to perform earlier stop training. Indeed, at 1000 epochs, MSE remains low, with the advantage of improving ANN generalization ability. **Figure 6** shows the PSD at this level.

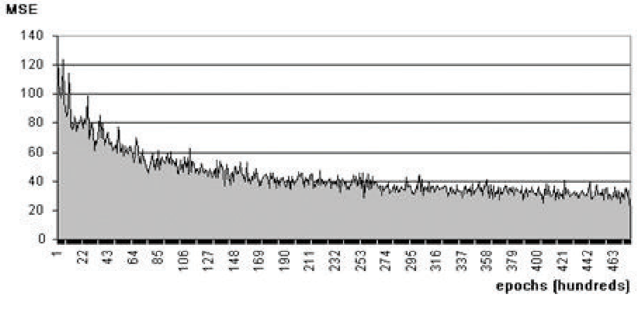


Figure 2. Evolution of the MSE expressed in hundred of epochs.

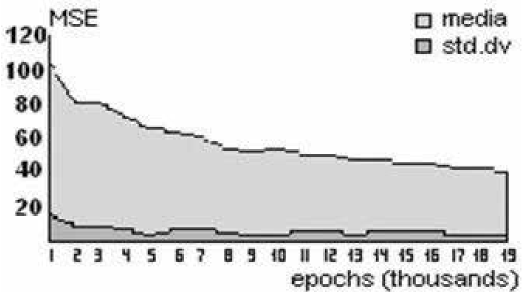


Figure 3. Evolution of media and standard deviation of the MSE.

Analyzing **Figures 5** and **6**, we observe that they are similar in medium potential order. It can be noted that the difference between predicted and real noise appears only in low frequencies. At the beginning of the training process, detrending is partial. Also, it can be concluded that the most trained ANN obtains more concordance between real and predicted noise spectral. At 30,000 epochs, differences are about 10 Hz, which is remarkable.

The results of this second approach based on ANNs methodology, showed good predictions for the Lorentz's function trend line, superimposed with Gaussian noise. It was observed that,

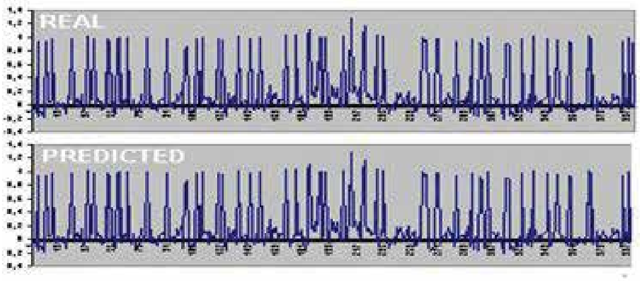


Figure 4. Comparison between real and ANN predicted noise.

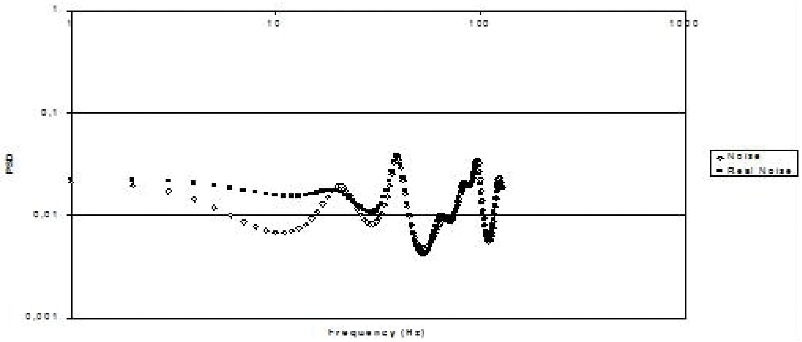


Figure 5. PSD at 30,000 iterations.

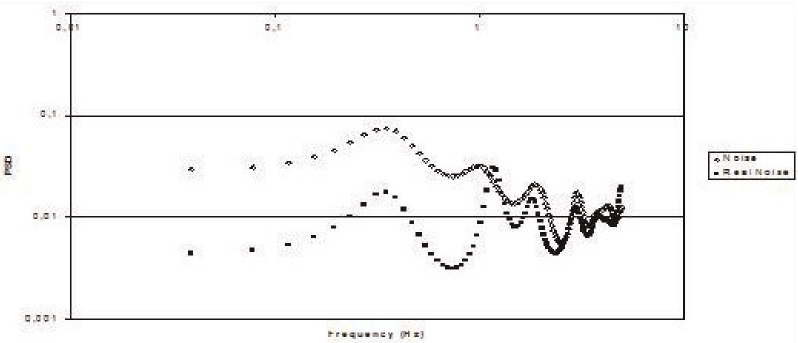


Figure 6. PSD at 1000 iterations.

especially at high frequencies, the superimposed signal has no influence when performing the ANN detrending. The corresponding results and their validation by PSD were published in a previous paper [24].

Nevertheless, poor results were obtained when the degrees of freedom were increased. This situation was particularly observed in low frequencies, when a non-Gaussian-simulated noise was superimposed. Moreover, if the difference between trend and noise increases, the results show that the method loses accuracy. As a consequence, the use of ANNs with real experimental data (having an unpredictable trend shape) would need a set of training curves or a previous selection of the network to be applied. Thus, a different approach for detrending was needed.

Even more, when analyzing experimental data, it is obvious that a Gaussian noise can be considered only as a preliminary approximation, since real curves show a different kind of noise consisting of nonstationary pits superimposed on the trend line.

For these reasons, our third approach to the problem had to include a different noise simulation in order to get a better approximation to experimental real-life curves.

5. New simulated signals

As in the previous cases, the signals to be simulated needed to be generated using different nontrivial tendencies. For this purpose, curves 1 and 2 of Section 2.1 were reutilized, but in this case, a new computer-generated noise was superimposed.

In this case, noise employed was a discrete transient of exponential decay that simulates a nonstationary pit. More precisely, the noise consisted of a pulse train with an amplitude factor (A) corresponding to a uniform distribution of [0, 2.5). Initial time was obtained from a binomial distribution with a parameter p=0.02 and its sign had a binomial distribution with parameter p=0.5. As a consequence of these facts, pits can appear randomly in 2% of the points, and it can be either positive or negative with the same probability.

The transients are simulated by the pulse function: $f(t) = A\,B\,t\exp\left(-b\,(t-c)^d\right)$. The parameters b, c, and d were assigned to obtain a signal similar to the experimental transient. The A factor represented the transient amplitude and B determined the sign of the function. Taking into account the previous description, the simulated pit—if it is positive—grows fast and then decays exponentially as it usually does in experimental curves. A similar behavior is presented in negative nonstationary pits.

As a final remark, another characteristic of the simulated transients is the similarity of format, frequency, and amplitude relative to the real signal with respect to real transients.

Figure 7 shows a detailed shape of individual simulated transitory.

Then, two computer-simulated data sets were generated by adding the new noise—described in this section—to both trend lines presented in Section 2. A first signal was obtained by adding the new noise function to the first curve of Section 2.1, and the same procedure was followed to obtain a second simulated signal, adding noise to the second curve of Section 2.2. In all cases, the interval between data points was 0.7 s.

Figure 8 represents the simulated pulse train and signals, where parameters were fitted in order to obtain curves with similar power trend lines.

Both curves represent more realistic simulations, since their shapes and pits are similar to those observed in the laboratory (this fact can be observed in [30], where simulated and experimental curves were compared).

The simulated Signal 1 (New noise superimposed to Curve 1) was treated with polynomial, MMR10, and MAR10 methods. Other methodologies previously studied as MICS, Butterworth

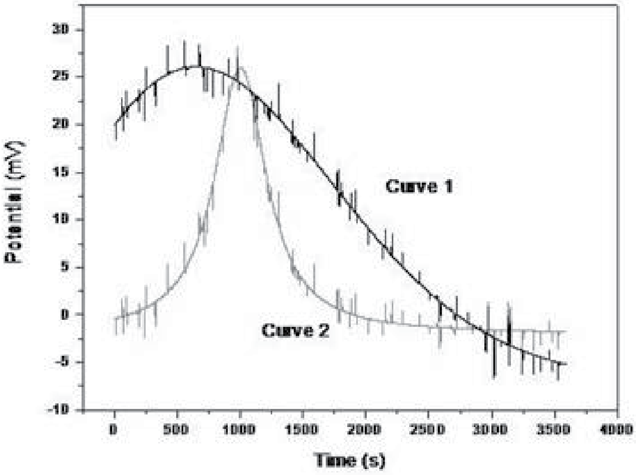


Figure 7. The simulated transitory detailed.

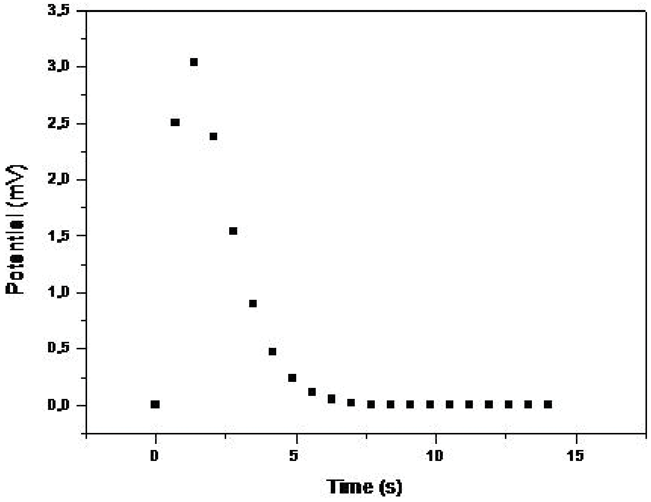


Figure 8. Computationally generated signals, obtained by adding simulated noise data and two different trend lines. Curve 1 (black) and Curve 2 (grey).

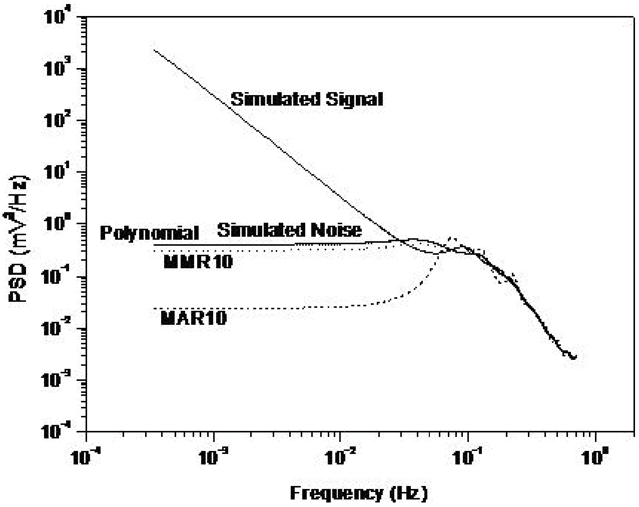


Figure 9. PSD of simulated Signal 1 (Noise + Curve 1), treated with polynomial, MMR10, and MAR10 methods.

filters, and ANNs were not considered at this stage as their results were very poor when they were assessed by using this new signal.

Figure 9 shows simulated Signal 1 and noise spectra, obtained by the maximum entropy method (MEM −15 order, ENAnalize® program).

Likewise, simulated Signal 1 (New noise superimposed on Curve 2) was treated with the same three detrending methods: polynomial approximation, MMR10, and MAR10. As in the

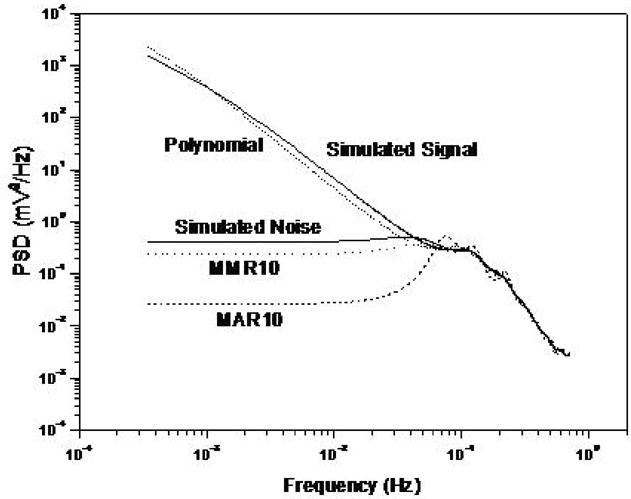


Figure 10. PSD of simulated Signal 2 (Noise + Curve 2) treated with polynomial, MMR10, and MAR10 methods.

previous case, MICS, Butterworth filters, and ANNs showed very poor results when working with this new signal, and so they were not considered.

Figure 10 shows the results corresponding to Signal 2.

The power spectrum obtained depends on the order of the MEM method applied. A smooth spectrum is observed when the MEM order is small, whereas the spectrum appears much noisier with a high order. The comparison between the spectra obtained by applying different detrending methods is easier when using a relatively low-order MEM [3]. More accurate results are provided by Fast Fourier Transform (FFT), in spite of the spectra obtained being noisy. Then, the comparison of PSD results in the same graphic is not plainly depicted. It is not possible to compare the PSD results for FFT with the MEM results, and, for this reason, they are not included in **Figures 9** and **10**.

6. A comparison of the performance of different detrending methods

The aim of this new approach was to analyze the performance of several detrending methods, when assessed by using the new signals with simulated nonstationary pits instead of Gaussian noise [29].

Due to their performance with these new simulated data, the selected methods were polynomial detrending, moving average removal (MAR), and moving median removal (MMR). The first two were described before and the third one can be considered similar to the second one, except for the substitution of the moving average by the moving median. In MMR-10 method, the noise is computed as $x_\mathrm{n} - k_\mathrm{n}$, whereas x_n denotes an experimental value and k_n represents the moving median:$k_\mathrm{n} = median\ \left[x_{\mathrm{n}-10}, \ldots, x_\mathrm{n}, \ldots x_{\mathrm{n}+10}\right]$.

		Signal 1	Signal 2
MAR-10	Mean	2.49×10^{-4}	-2.93×10^{-4}
	Standard deviation	0.3204	0.3204
MMR-10	Mean	8.50×10^{-3}	5.90×10^{-3}
	Standard deviation	0.3600	0.3550
Polynomial	Mean	-1.39×10^{-5}	4.13×10^{-3}
	Standard deviation	0.3735	1.8261

Table 1. Mean and standard deviation for trend removal methods: MAR-10, MMR-10, and ninth-order polynomial fitting, under simulated pulse train noise (mean: −0.0112, standard deviation: 0.3736).

The simulated noise data set had a mean of −0.0112 and standard deviation of 0.3736. **Table 1** shows the statistical results for the trend removal methods employed for simulated Signal 1 and Signal 2.

The noise mean value reported was close to zero with all the methods performed (**Table 1**), so it is not possible to conclude about the performance of detrending methods considering only this parameter.

- Ninth-order polynomial fitting

In the case of Signal 1 (with a smoother trend), **Table 1** shows that the standard deviation is similar to the one of the original noise, while a greater standard deviation was obtained for Signal 2.

This behavior is confirmed by the spectra analysis, since the polynomial methodology does not remove the power in the low-frequency region for Signal 2. Moreover, the spectrum obtained for Signal 1 is a good representation of the trace of the original simulated noise. Thus, this detrending method strongly depends on the simulated trend curve, and it is not reliable as a pre-processing method.

- MAR-10

This method does not depend on the trend curve utilized (**Table 1**). Indeed, the data standard deviation showed a good agreement with the simulated noise. Examining the spectrum, we can affirm that at high frequencies the noise obtained fits well with the original noise. On the other hand, for frequencies below 0.01 Hz, there was a maximum at ca. 0.07 Hz and a plateau was reached, with smaller power values than the original noise. A frequency breakpoint value was obtained after processing the original noise spectrum data with MAR compared to the one predicted by the interval obtained by using the transfer function. Bertocci et al. [13] presented the MAR-p transfer function, as can be seen below:

$H_{MAR}(f) = 1 - \frac{1}{2p+1}\frac{\sin\left([2p+1]\pi\frac{f}{f_s}\right)}{\sin\left(\pi\frac{f}{f_s}\right)}$, being f_s the sample frequency.

The spectrum is not attenuated between $f_{\max}/(2p + 1)$ and $f_{\max}$. The highest representative frequency is the Nyquist frequency ($f_{\max} = f_s/2$). In the case considered, the interval used was between 3.4×10^{-2} and 0.7 Hz.

In sum, a lower standard deviation was obtained by the spectrum recovered by MAR-10 than the one related to the original data.

- MMR-10

MMR and MAR methods were robust (considering the type of signal processing), when compared with polynomial fitting. MMR standard deviation was much closer than the one obtained by MAR respect to the original values. A lower difference between the original value and the one reached by MMR was obtained for Signal 1. The MMR-10 spectrum fits better with the original noise than the one obtained by MAR-10 in all the frequency range. These results were discussed in-depth in a previous paper [29].

The mean is the most commonly used statistic measure of location in corrosion processes. However, extreme values will have great influence on it. The average may not be the most appropriate location measure if there are outliers in the sample, that is, if one or more values are much larger or smaller than the others [30].

On the one hand, data must be ordered increasingly for calculating a median, and for a large data set, this is a time-consuming task and so calculating medians are computationally more demanding than calculating averages. On the other hand, medians remain unaffected by a small group of outliers [29, 30]. This is a very important characteristic for approximating the trend in an ENA data set. For these reasons, MMR provides a better baseline than MAR, since the subtraction of medians preserves the signal trace, showing less attenuation than the corresponding subtraction of the average.

7. Conclusion

As observed, the method used for trend removal affected the results obtained. An ideal methodology should not introduce external effects, and at same time, it should recover most of the signal information.

The polynomial fitting method showed a strong dependence when different trends were considered. This methodology presented good results for smooth tendencies and poor performances when curves changed suddenly in relation to their slope and convexity.

When working with different curves, MAR presented robust results. Nevertheless, alterations in the low-frequency zone of spectra were observed, and as a consequence of this fact, the standard deviation results were not accurate.

Besides, MMR, our proposed method, demonstrated a better performance when working with different simulated signals, although in the low-frequency zone of spectra, it showed the same behavior as MAR.

Taking into account the results obtained with simulated data, it is difficult to arrive to definitive conclusions. As it was observed, the polynomial fitting method is not reliable for trend removal at least when working with ENA data. On the other hand, methods like MAR and MMR achieve better results, but they do not offer a definitive solution. In fact, the results suggest that the application of these methods produces an apparently attenuated response with low noise signals.

Hence, MMR method seems to show the best results. Nevertheless, increasing the order of magnitude, this method did not improve its performance, and as previously observed, calculating the median is computationally more troublesome than calculating the average of a given data set.

As it was shown with the new simulated signals and, despite all the studied methods have been useful in many particular cases, a really accurate procedure for trend removal is still an open problem, giving new opportunities for further research.

Acknowledgements

The authors thank Marjorie Chaves for the assistance with the revision of the manuscript.

Author details

Victor Martinez-Luaces* and Mauricio Ohanian

*Address all correspondence to: victorml@fing.edu.uy

Faculty of Engineering, UdelaR, Montevideo, Uruguay

References

[1] Almeida E, Mariaca L, Rodriguez A, Uruchurtu J, Veloz M. Characterization of prerusted steels in some Ibero-American atmospheres by electrochemical potential noise measurement. In: Kearns J, Scully J, Roberge P, Reichert D, Dawson J, editors. Electrochemical Noise Measurement for Corrosion Applications. West Conshohocken, USA: ASTM International; 1996. pp. 411-426. DOI: 10.1520/STP37974S

[2] Dawson J. Electrochemical noise measurement: The definitive in-situ technique for corrosion applications? In: Kearns J, Scully J, Roberge P, Reichert D, Dawson J, editors. Electrochemical Noise Measurement for Corrosion Applications. West Conshohocken, USA: ASTM International; 1996. pp. 3–38. DOI: 10.1520/STP37949S

[3] Cottis R. Interpretation of electrochemical noise data. Corrosion. 2001;**57**(3):265-285. DOI: 10.5006/1.3290350

[4] Sachs L. Applied Statistics: A Handbook of Techniques. 2nd ed. New York: Springer-Verlag; 2012. 707 p. DOI: 10.1007/978-1-4612-5246-7

[5] Hetch-Nielsen R. Neurocomputing. Massachusetts: Addison-Wesley; 1990. 433 p

[6] Font C, Manrique D, Ríos J. Redes de Neuronas Artificiales y Computación Evolutiva. Madrid: Universidad Politécnica de Madrid; 2005. 239 p

[7] Koza J, Rice J. Genetic generation of both the weights and architecture for a neural network. In: Proceedings of Seattle International Joint Conference on Neural Networks (IJCNN-91); 8-12 July 1991; Seattle IEEE; 1991. pp. 397-404

[8] Ohanian M, Martinez-Luaces V, Guineo G. Highly dispersed electrochemical noise data: Searching for reasons and possible solutions. The Journal of Corrosion Science and Engineering. 2004;**7**:preprint 6

[9] Ralston A, Rabinowitz P. First Course in Numerical Analysis. 2nd ed. New York: Dover Publications; 2001. 606 p

[10] Chan K, Hayya J, Ord J. A note on trend removal methods: The case of polynomial regression versus variate differencing. Econometrica: Journal of the Econometric Society. 1977;**45-3**:737-744. DOI: 10.2307/1911686

[11] Mansfeld F, Sun Z, Hsu C, Nagiub A. Concerning trend removal in electrochemical noise measurements. Corrosion Science. 2001;**43**(2):341-352. DOI: 10.1016/S0010-938X(00)00064-0

[12] Tan Y, Bailey S, Kinsella B. Factors affecting the determination of electrochemical noise resistance. Corrosion. 1999;**55**(5):469-475. DOI: 10.5006/1.3284009

[13] Bertocci U, Huet F, Nogueira R, Rousseau P. Drift removal procedures in the analysis of electrochemical noise. Corrosion. 2002;**58**(4):337-347. DOI: 10.5006/1.3287684

[14] Ohanian M, Caraballo R, Dalchiele E, Guineo-Cobs G, Martínez-Luaces V, Quagliata E. Productos de corrosión formados en ambiente marino: Vinculación de variables estructurales y potencial de corrosión. Revista de Metalurgia. 2005;**41-3**:175-185. DOI: 10.3989/revmetalm.2005.v41.i3

[15] Ohanian M. Corrosión Atmosférica: Caracterización estructural y electroquímica de productos de corrosión formados sobre acero de baja aleación en intemperie marina [Thesis]. Montevideo, Uruguay: Universidad de la República; 2004

[16] Faires J, Burden R. Numerical Methods. 4th ed. Boston: Brooks/Cole; 2012. 608 p

[17] Hilera J, Martínez V. Redes Neuronales Artificiales: Fundamentos, Métodos y Modelos. Editorial Alfaomega: México; 2000. 390 p

[18] Stergiou C, Siganos D. Neural Networks [Internet]. 2010. Available from: http://www.doc.ic.ac.uk/~nd/surprise_96/journal/vol4/cs11/report.html [Accessed: 2017-12-26]

[19] Haykin S. Neural Networks: A Comprehensive Foundation. New Jersey: Prentice-Hall; 1999. 842 p

[20] Martinez Luaces M. Metodologías en Redes Neuronales: Diseño, Entrenamiento y Validación de Perceptrones Multi-Capa. In: Proceedings of COMPUMAT; La Habana; 2003

[21] Martinez Luaces M, Martinez-Luaces V. Teachers assessment: A comparative study with neural networks. In: Proceedings of DELTA Conference (DELTA'03); 23–27 November 2003; Queenstown. Dunedin: ISC-DELTA; 2003. pp. 180-185

[22] Hassoun M. Fundamentals of Artificial Neural Networks. Massachusetts: MIT Press; 1995. 511 p

[23] Picton P. What is a neural network? In: Introduction to Neural Networks. London: Palgrave; 1994. pp. 1-12. DOI: 10.1007/978-1-349-13530-1_1

[24] Martinez Luaces M, Martinez-Luaces V, Ohanian M. Trend-removal with neural networks: Data simulation, preprocessing and different training algorithms applied to electrochemical noise studies. In: Proceedings of the 5th WSEAS International Conference on Artificial Intelligence, Knowledge Engineering and Data Bases; 15-17 February 2006; Madrid. Madrid: WSEAS; 2006. pp. 810-817

[25] Freeman J, Skapura D. Neural Networks: Algorithms, Applications and Programming Techniques. Massachusetts: Addison-Wesley; 1991. 401 p

[26] Salvetto P, Martínez Luaces M, Luna C, Segovia J. A very early estimation of software development using neural network. In: Proceedings of X Congreso Argentino de Ciencias de la Computación (CACIC'10); 4-8 October 2004; San Justo. Buenos Aires: CACIC

[27] Isasi P, Galván I. Redes de neuronas artificiales: Un enfoque práctico. Pearson Education: Madrid, Spain; 2004. 229 p

[28] Ventura D, Andersen T, Martinez T. Using evolutionary computation to generate training set data for neural networks. In: Proceedings of the International Conference on Artificial Neural Networks and Genetic Algorithms; 19-21 April 1995; Alès, France. Berlin: Springer; 1995. pp. 468-471

[29] Ohanian M, Martinez-Luaces V, Diaz V. Trend removal from electrochemical noise data. The Journal of Corrosion Science and Engineering. 2010;**13**:preprint 52

[30] Utts J, Heckard R. Mind on Statistics. 5th ed. Cengage Learning: Stamford; 2014. 682 p

Edited by M. Serdar Çelebi

Computational science and engineering (CSE) is a broad multidisciplinary and integrative area including a variety of applications in science, engineering, numerical methods, applied mathematics, and computer science disciplines.

The book covers a collection of different types of applications and visions to various disciplinary key aspects, which comprises both problem-driven and methodology-driven approaches at the same time. These selected applications are:

- Computational and information technologies for numerical models and large unstructured data processing
- Evolution of matrix computations and new concepts in computing
- Inverse problems covering both classical and newer approaches
- Integro-differential scheme (IDS) that combines finite volume and finite difference methods
- Smart city wireless networks
- Signal processing methods

Published in London, UK

IntechOpen

ISBN 978-1-83881-470-0
9 781838 814700