95-th European Study Group with Industry (ESGI'95)

September 23 – 27, 2013
Sofia, Bulgaria

PROBLEMS
&
FINAL REPORTS


Demetra
2013

Organizers:

Faculty of Mathematics and Informatics,
Sofia University "St. Kl. Ohridski",

in cooperation with

Institute of Information and Communication Technologies, BAS and

Institute of Mathematics and Informatics, BAS

with the kind support of

European Consortium for Mathematics in Industry and

Oxford Center for Collaborative and Applied Mathematics

Scientific Advisory Committee:

Prof. Dr.Sc. Ivan Soskov, Dean of FMI, SU,
Prof. Dr.Sc. Svetozar Margenov, Director of IICT, BAS
Prof. Dr.Sc. Julian Revalski, Director of IMI, BAS,
Prof. Wil Schilders, ECMI Board.

Local Organizing Committee:

Chairperson: Doc. Dr. Tatiana Chernogorova, FMI, SU
Prof. Dr. Maria Nisheva, Vice-Dean of FMI, SU,
Doc. Dr. Nadya Zlateva, Vice-Dean of FMI, SU,
Prof. Dr.Sc. Stefka Dimova, Dept. Num. Methods and Algorithms, FMI, SU,
Doc. Dr. Krasimir Georgiev, IICT, BAS
Doc. Dr. Neli Dimitrova, IMI, BAS,
Ass. Prof. Ivan Hristov, Dept. Num. Methods and Algorithms, FMI, SU.

Published by Demetra Publishing House, Sofia, Bulgaria
ISBN 978-954-9526-84-4

Contents

Preface

The 95th European Study Group with Industry (ESGI'95) held in Sofia, Bulgaria, September 23–27, 2013. It was organized by the Faculty of Mathematics and Informatics, Sofia University "St. Kl. Ohridski" (FMI, SU) in cooperation with the Institute of Information and Communication Technologies, Bulgarian Academy of Sciences (IICT, BAS) and the Institute of Mathematics and Informatics, BAS (IMI, BAS).

This first for Bulgaria Study Group with Industry was initiated by the European Consortium for Mathematics in Industry (ECMI) and strongly encouraged and assisted by Prof. Wil Schilders, a former president of ECMI. We express our gratitude to Prof. Schilders for his invaluable help and participation at ESGI'95. We express special thanks to Dr. Hilary Ockendon, administrative director of ECMI, for her help before and during ESGI'95 meeting. We also acknowledge her valuable contributions to the work on Problem 6.

ESGI'95 was partially supported by the Sofia University grant N98/2013. The partial financial support of ECMI and of the Oxford Center for Collaborative and Applied Mathematics (OCCAM) are also highly appreciated.

ESGI'95 was hosted by the Institute of Mathematics and Informatics, BAS and by the Institute of Information and Communication Technologies, BAS. The two institutions have provided excellent conditions for work.

Study Groups with Industry are an internationally recognized method of technology transfer between the academic mathematical community and the industry. These one week long workshops provide a forum for industrial scientists to work alongside academic mathematicians on problems of direct industrial relevance.

Six problems proposed by five Bulgarian companies were chosen by the Organizing Committee:

1. Adiss Lab Lts.: Compressed representation of a partially defined integer function over multiple arguments;

2. Adiss Lab Lts.: Estimation of errors in text and data processing;

3. KCM AD: Optimization of the charging process in zinc hydrometallurgy;

4. Mini Max: Mini Max Wallpaper;

5. MM Solutions AD: Laboratory calibration of a MEMS accelerometer sensor;

6. ppResearch Ltd.: Prediction of sanding in subsurface hydrocarbon reservoirs.

Thirty eight participants from Bulgaria (30) and from abroad (8) were divided into six working groups (teams). The work in each group was guided and consulted by a company representative that provided the problem. The Bulgarian participants were from various academic institutions: FMI, Sofia University (13); IMI, BAS (8); IICT, BAS (4); Plovdiv University (2); Veliko Tarnovo University (1); Technical University of Sofia (2). Sixteen of the participants were students: 1 PhD, 12 Master, 3 last year Bachelor students.

The progress in solving the problems and recommendations for possible further improvements were presented at the last day of the workshop. Further, each team presented a report containing the obtained results, comments, ideas, progress, and suggestions on their problem. These reports will provide a formal record for both the industrial and academic participants. The reports of all teams are assembled in this booklet to form the Study Group Final Report.

The description of the problems, the Workshop last day presentations and the final reports of each team are posted on the website of the ESGI'95:
`http://esgi95.fmi.uni-sofia.bg`

We believe that the main objectives of the SGI – to identify areas for cooperation between the Bulgarian academic community and the Bulgarian industry and to create contacts between the researchers, educators and developers (from companies) that may result in future funding on a national level – were definitely achieved. Encouraged by this success, we intend to make the Study Groups with Industry an every year tradition.

List of participants

Alexander Peltekov (KCM AD/Plovdiv University, FMI)
Angela Slavova (IMI, BAS)
Antonio Marigonda (University of Verona, Italy)
Borislav Valkov (Sofia University, FMI)
Clemens Hofreither (IICT, BAS/JKU Linz, Austria)
Daniela Vasileva (IMI, BAS)
Doychin Boyadzhiev (Plovdiv University, FMI)
Dragomir Aleksov (Sofia University, FMI)
Hilary Ockendon (Oxford University, England)
Irina Georgieva (IMI, BAS)
Ivan Georgiev (IMI, BAS)
Jan Idziak (Wroclaw University of Technology, Poland)
Kiril Aleksiev (IICT, BAS)
Krasimir Tonchev (Sofia University/Technical University of Sofia)
Krasimir Georgiev (Sofia University, FMI)
Margarita Nikolova (Sofia University, FMI)
Mariusz Kozlowski (Wroclaw University of Technology, Poland)
Michail Todorov (Technical University of Sofia, FAMI)
Mikhail Krastanov (Sofia University, FMI)
Milena Veneva (Sofia University, FMI)
Mira Bivas (Sofia University, FMI)
Monika Sikora (Wroclaw University of Technology, Poland)
Natalia Kolkovska (IMI, BAS)
Nina Daskalova (Sofia University, FMI)
Plamen Mateev (Sofia University, FMI)
Roumyana Yordanova (IMI, BAS)
Slav Angelov (Sofia University, FMI)
Stanislav Stoykov (IICT, BAS)
Stefka Dimova (Sofia University, FMI)
Stela Zhelezova (IMI, BAS)

Stoyan Poryazov (IMI, BAS)
Svetoslav Nakov (Sofia University, FMI)
Teodora Ilieva (Sofia University, FMI)
Tihomir Ivanov (IMI, BAS)
Tomasz Zorawik (Wroclaw University of Technology, Poland)
Wil Schilders (TU Eindhoven, Netherlands)
Zlatko Varbanov (University of Veliko Tarnovo)
Zlatogor Minchev (IICT, BAS/IMI, BAS)

PROBLEMS

Problem 1. Compressed representation of a partially defined integer function over multiple arguments

Adiss Lab Lts.

Zhivko Angelov

Problem description

In OLAP (OnLine Analitical Processing) data are analysed in an n-dimensional cube. The cube may be represented as a partially defined function over n arguments

$$f(x_1, x_2, \ldots, x_n), \quad \text{where} \quad x_i \in [1 \div N_i] \quad \text{for} \quad i \in [1 \div n].$$

In the general case the values of f are real numbers but quite often they are whole numbers (integers). Most commercial applications store values as fixed precision numbers, so if we assume that f is an integer function this will encompass a significant part of the applications.

Let $(*, *, \ldots, a_i, *, \ldots, *)$ denote an $(n-1)$-dimensional cube. The most frequently solved problem in OLAP is the computation of the sum of values F of a given sub-cube of the n-dimensional cube. In the case of the $(n-1)$-dimensional cube the values F are:

$$F(*, *, \ldots, a_i, *, \ldots, *) =$$

$$\sum_{j_1}^{N_1} \sum_{j_2}^{N_2} \cdots \sum_{j_{i-1}}^{N_{i-1}} \sum_{j_{i+1}}^{N_{i+1}} \cdots \sum_{j_n}^{N_n} f(j_1, j_2, \ldots, j_{i-1}, a_i, j_{i+1}, \ldots, j_n)$$

Other frequently computed functions beside summation are arithmetic average, minimum and maximum.

Question 1: Considering that often the function f is not defined everywhere, is there a known way of representing f or the points, in which it is defined, in a more compact manner than the trivial $a_1, a_2, \ldots, a_n$, $f(a_1, a_2, \ldots, a_n)$? The goal is to reduce the time necessary for moving or storing f and using part of the time gained for computations for restoring the original f.

Question 2: Due to natural limitations the visualization of a n-dimensional cube is reduced to the representation of a rectangle in a particular plain.

One possible approach is to compute the values of all sub-cubes

$F(*, x_2, x_3, \ldots, x_n)$,

$F(x_1, *, x_3, \ldots, x_n)$,
$\ldots$
$F(*, *, x_3, \ldots, x_n)$,
$\ldots$
$F(*, *, \ldots, *)$
and to show the values of cubes located in the defined rectangle.

In this case the transition from one rectangle to another in the considered plain and the transition to a plain another dimension will be accomplished in approximately one and the same short time interval (includes only the reading of several hundred values). This is in contrast to the time needed for the computation of the values and the memory necessary for their storage which may be significant.

In the visualization process only a subset of set of all sub-cubes is used, only the values of these cubes are necessary. How can an algorithm be constructed that makes a balance between preliminarily computed values and the computation of values that are not stored but fall into the defined visualization area? If there exists a compressed representation of data (Question 1) is it possible to base the algorithms on this representation? Considering the fact that in many applications the function f is of integer type do more efficient algorithms exist under the restriction of f being integer?

An important fact that has to be taken into account is that almost always the values of the whole cube take part in the visualization process as well as the values of several sub-cubes (as illustrated below). This means that these values have to be computed before the visualization start.

Arrivals

Facts

Arr Date	Country	Airport: Plovdiv	Sofia	Total
05.09.2013			1	1
06.09.2013		1	17	18
07.09.2013		1	10	11
08.09.2013		2	46	48
09.09.2013	Belgium		1	1
	England		1	1
	France		1	1
	Germany		1	1
	Luxembourg		1	1
	Sfax- Tunisia		1	1
	South Africa		1	1
	Spain		1	1
	Total		8	8
10.09.2013			1	1
11.09.2013			4	4
12.09.2013			2	2
Total		4	89	93

Problem 2. Estimation of errors in text and data processing

Adiss Lab Lts.

Zhivko Angelov

Problem description

Much important information today is stored in free text format. This includes patient-related records which contain essential clinical facts related to the patient status and treatment. The recent achievements in automatic text analysis enable extraction of specific entities and/or events with high accuracy. In this way it becomes possible to plan large-scale initiatives in recognition of particular entities and storing them in data bases for further processing by data analytics software. This process is illustrated in Fig. 1: separate text descriptions are analysed, in order to structure their content, and the resulting data base is subject of further mining for statistical purposes.

Typically, the success of Information Extraction (IE, a kind of partial automatic text analysis) is measured by:

- ***Precision***: the number of correctly extracted entity descriptions, divided by the number of all recognised entity descriptions in the test set;
- ***Recall***: the number of correctly extracted entity descriptions, divided by the number of all available entity descriptions in the test set (some of them may remain unrecognised by the particular IE module).

Thus the Precision measures the success and the Recall – the recognition ability and "sensitivity or coverage" of the algorithms. The F-score (harmonic mean of Precision and Recall) combines the two measures and is defined as $F = 2 \times Precision \times Recall/(Precision + Recall)$.

In order to support the IE tasks, various types of language resources are incrementally developed since decades. This includes corpora used as training or test data sets as well as "gold standard" annotated corpora that enable comparisons of different software systems.

Questions:

1. Million Patient records exist, typed in by thousands medical experts. In this way automatic assessment of the IE correctness by mapping the extracted entities to certain "gold standard" is impossible. Let

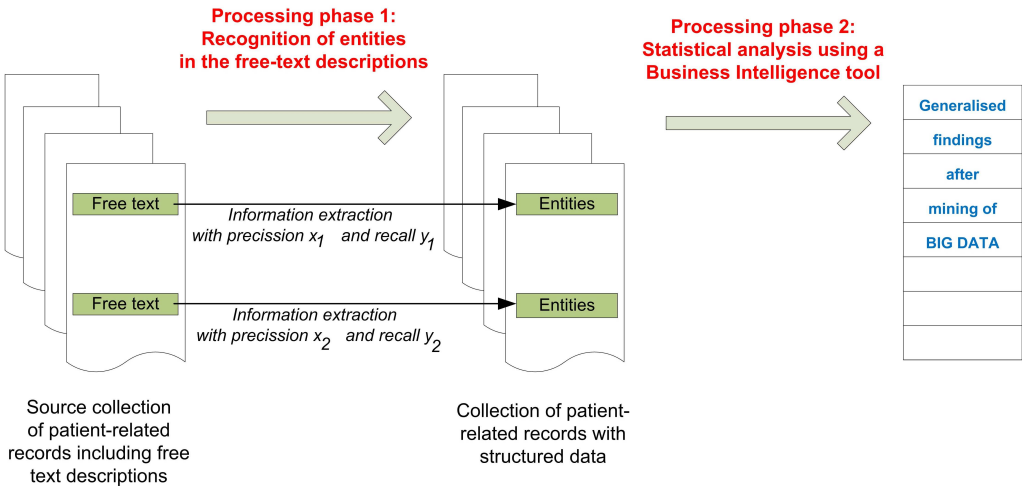


Fig. 1. Phases in text analytics: extraction of specific entities and further data mining

- the source collection of documents contains N records,
- one kind of focal event is DRUG THERAPY where an IE module extracts from the text drug codes, dosage, frequency and admission route, for 1500 drugs,
- another kind of extracted events are LAB TESTS where another IE module extracts from the text the Lab test type and the particular value, for 800 kinds of lab tests.

How many records from the collection with the structured data have to be inspected manually, in order to assess the precisions x_1 and x_2 and the recall y_1 and y_2?

2. Could you please suggest some measure Z for the "overall correctness" of the collection with the structured data, which integrates the results of several (at least two) independent IE modules?

3. What should be the minimal correctness Z for a collection with N records, to claim, that the data mining in phase 2 delivers credible observations?

Problem 3. Optimization of the charging process in zinc hydrometallurgy

KCM AD

Alexander Peltekov

Problem description

KCM AD is part of the holding structure of KCM 2000 AD, Plovdiv. The company is a leading producer of nonferrous and precious metals in Southeast Europe and the Black Sea Region, working actively for the environmental recognition of metals as efficient market products. KCM's annual production capacity of lead is 65 000 t., and of zinc – 80 000 t.

The charging is a process of mechanically mixing raw materials in various quantities to obtain suitable charge for subsequent processing (in hydrometallurgy – a mixture of the different zinc sulfide concentrates). The charge is considered as suitable if it satisfies some restrictions on the production technology and organization. The production technology determines the chemical composition of the charge (useful and harmful components in the production process), and the organization – the route of dosing (quantization) at mixing of various raw materials.

One possible criterion for optimization is to determine the suitable charge with minimal cost. The stages for solving the task are:

1. Creating a database for chemical composition of zinc sulfide concentrates in KCM AD (about 20 components of a concentrate) and restrictions associated with the technology of hydrometallurgical production.

2. Determination of "technology" cost of individual concentrate (by a given algorithm, depending on its composition and the current price of zinc in the international markets – `www.lme.co.uk`).

3. Formulation of linear optimization model for charge with minimum 'technology' price.

4. Formulation of discrete optimization problem, taking into account the particular organization of production.

5. Choice of software to solve the problems.

6. Numerical experiments and discussion of results.

Problem 4. Mini Max Wallpaper

Mini Max

Elena Hekimova

Mini Max is a company specialising in the distribution and retail sale of wall-coverings. In order to facilitate our clients, we need an automated way to calculate the number of wallpaper rolls necessary for decorating a room with wallpaper. The final calculation will be implemented as part of a web-based calculator open for use to both Mini Max staff and the general public.

Room Measurements

Clients provide the measurements of their rooms: number of walls to be decorated with wallpaper, wall width and height, and sizes of doors and windows.

Wallpaper Specifications

There are several possible sizes of wallpaper rolls:

0.53 m × 10 m

0.70 m × 10 m

1.06 m × 10 m

1.06 m × 25 m

Wallpaper designs can be plain or patterned. Plain wallpapers are made of just one colour and do not need to be adjoined at the sides when applied to the wall. Patterned wallpapers have a design, which requires the wallpaper to be matched at the sides. There are two match types: straight match (see figure A) and offset match (see figure B). Straight match wallpaper starts with the same pattern at the top of each wallpaper strip, offset match wallpaper starts with the same pattern at the top every other strip. Each wallpaper design has a stated by the manufacturer pattern height, and, if applicable in the case of offset match, a drop height. The drop height is the length by which the pattern is dropped at the alternate strips of wallpaper.

The calculation of the number of necessary rolls should take into account the pattern specifics of each wallpaper design.

Below is a table with sample data for different wallpaper designs:

Fig. A (Straight Match)

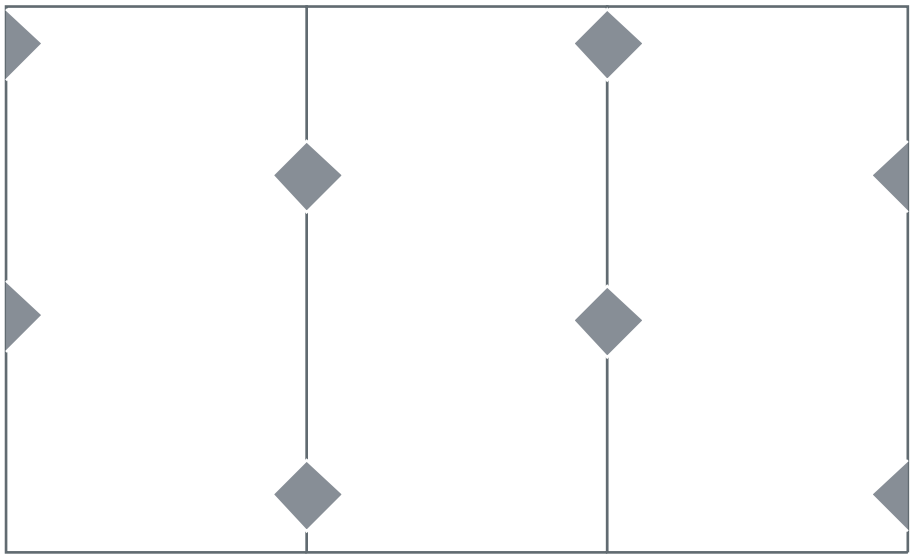


Fig. B (Offset Match)

Match Type	Pattern Height (cm)	Drop (cm)
straight	64	–
straight	53	–
straight	45	–
straight	4	–
straight	5	–
straight	1.5	–
straight	16	–
straight	1	–
straight	21.5	–
straight	91	–
straight	32	–
offset	64	32
offset	26.5	13.25
offset	45	22.5
offset	32	16
offset	53	26.5
free	0	0

It should be taken into account that only full strips are used for a full room height, e.g. if the height of a wall is 2.6 m, it is necessary to use one piece of wallpaper which is 2.6 m long, it is not possible to use 2 pieces of 1.3 m since matching may not be possible and there will be a horizontal matching point which will be visually unpleasant. Leftover pieces of wallpaper may, however, be used around doors and windows if they match correctly. In addition, when cutting the wallpaper to size for the required wall height, it is necessary to take into account a 10 cm safety margin, which is used in case the ceiling or floor is not completely straight.

The web-based calculator will require the user to fill out the room measurements, as well as the doors and windows. The data for each wallpaper design (pattern height, type of pattern, roll size) is stored in our wallpaper database and is queried depending on the wallpaper the user selects. The calculator should then output the necessary number of rolls for the specific room and the selected design. The task is to develop a calculation method that takes into account all the room and wallpaper specifics.

Problem 5. Laboratory calibration of a MEMS accelerometer sensor

MM Solutions AD

Angel Ivanov

Background

MEMS accelerometer and gyroscope sensors became a commodity in the mobile devices. Allowing to measure the acceleration and the rotational speed in all directions, they enable many applications to benefit from the ability to estimate the device pose and motion. In conjunction with the build-in camera, novel features can be offered to the end user like navigation, guidance in a large or small environment, detection of objects of interest, augmented reality, real terrain video games, and many others.

The specific use case for this study is to use the accelerometer sensor along with a gyroscope and a camera, for estimating and tracking the camera position and orientation in space. Since the MEMS gyroscope sensors are cheap and low quality, the production tolerances play a significant role and must be taken onto account.

Constraints

Accelerometer sensors measure the linear acceleration (or actually the force) in certain direction called axis. A combination of 3 accelerometers in a single chip with orthogonal axes is used in the mobile devices, giving the ability to calculate the device position (the camera in this case) in the 3D space. Since the device may rotate, the measurement axes change their directions in space. Thus, the camera rotation should first be calculated using the gyroscope sensor.

The gravity applies also a force, which is measured as acceleration when the device is still. This gravity acceleration adds to the true acceleration, caused by the real motion which is to me measured, so it has to be subtracted from the accelerometer data. The gravity acceleration is quite big compared to the normal motion accelerations, so knowing exactly the gravity direction and magnitude is vital for achieving useful pure motion acceleration.

There are a few other factors that affect the accuracy of the measurement, which are related to the technology used itself.

- offset

  Even in still condition, the accelerometer outputs some value, called offset, meaning some fake acceleration. It has to be known and subtracted from the data before using it for calculations.

- scale

  "Scale" means the measurement units of the accelerometer sensor – those are usually m/s^2, but the actual measurement unit may deviate proportionally around this value. Improper scale will result in scaling all the calculated camera positions, and (which is more fatal) improper cancellation of the gravity acceleration, resulting in counting a fake acceleration.

- crosstalk or non-orthogonality

  In a perfect triple accelerometer, acceleration on one of the axes should not cause any measurement on the other axes. Unfortunately this is not the case in practice. Part of the measured acceleration in the "right" axis appears in the other axes too. This effect is known as crosstalk and its maximum amount is verified and specified by the vendors in the sensor specifications. The reason for the crosstalk may be either electrical transfer in the chip or deviation from the perfect orthogonality of the sensors' axes. The most severe consequence is as in the case of improper scale, improper cancellation of the gravity acceleration.

The upper described parameters has to be defined in a mathematical model, resulting in a formulae for correction of the accelerometer data before being used in calculations for camera position estimation. Then, a calibration procedure should be defined, incorporating certain measurements and an algorithm for processing the measured data to achieve the calibration parameters of the defined mathematical model.

Another restriction is the inability to take the accelerometer sensor off the device. It is mounted in a mobile phone and may not be even visible. The calibration measurements should be captured using an application for a mobile phone for reading the accelerometer data directly on the device the accelerometer is mounted on.

The calibration procedure should not require expensive equipment and the whole calibration procedure of one accelerometer should be executable within 1 hour.

Potential Solutions

Many articles deal with the problem of mutual calibration of an accelerometer and other sensors (e.g. camera). Different approaches are investigated, including run-time calibration, i.e. refining the calibration parameters during the normal operation of the device. Those methods, however are quite computationally expensive, require long time for development and assume some initial calibration of the sensors.

A simple and quick solution for laboratory-based calibration is needed, that to be used as a first-stage calibration, followed by a run-time calibration, which should cover the temperature and aging deviations of the calibration parameters.

Aims for the study

The aim of this study is to define a suitable formulae for correction the accelerometer data and a simple and quick method for determining the parameters of this correction using a mobile phone with embedded accelerometer.

Sample data will be sent, measured from mobile phone. More data can be sent if needed. More data could be captured using any suitable mobile phone and a common application for reading and showing the accelerometer data.

Problem 6. Prediction of sanding in subsurface hydrocarbon reservoirs

ppResearch Ltd.

Peter Popov

Background

We are a start-up company which targets the hydrocarbon reservoir engineering community. We develop software tools for simulation of subsurface flows in deformable porous media. This is being done by modeling the physical problem by a system of partial differential equations. This system is discretized numerically and solved at a number of time steps. As a result of a simulation one has, as primary variable, the mechanical displacements and the fluid pressure resolved at any spatial location and time instance. A number of derived quantities, such as mechanical stresses, fluid velocity, etc are also computed. We are interested to relate such computations to localized processes which occur in particular locations in a subsurface reservoir.

The problem

Of primary interest is the nature of the solution near wellbores. These are the locations in a reservoir model from where fluids are injected/extracted. Locally, many physical changes may occur, which cannot be represented within the general framework of coupled poro-mechanics that is used to generate a solution, globally, that is in the entire reservoir. Global solutions have a characteristic lengthscale of tens of meters, while wellbores have radius of a few centimeters to tens of centimeter. Consequently, the global solution that is computed need to be appropriately downscaled near wellbores to help understand the local physics. We are interested in the local phenomena of sanding. Near wellbores the fluid velocity can become quit high, effecting high friction on the porous rock. Moreover, due to drilling operations the mechanical stresses can exceed the load bearing capacity of the rock. As a result, the rock can be fractured into sand, which is then carried by the fluid through the wellbore. This process, known as sanding, can severely and negatively impact the production at given well. Sand is abrasive to equipment, can clog flow lines and diminish hydrocarbon recovery. In a worst case scenario, sanding can remove sufficient quantities of rock to produce subsurface cavities which then collapse, destroying the well and associated equipment.

As a rule of thumb, every effort is made to avoid the conditions which lead to sanding. When sanding is unavoidable it is necessary to estimate the magnitude of the problem. What we would like to do, as part of this study group is to generate a simple one-dimensional local model which predicts the volume of sanding based on stresses and fluid velocities near a well. To do this, we take certain values for globally computed fluid pressure and mechanical stresses. We then exploit the radial symmetry of a well to compute localized stresses and pressure at the wellbore surface. These are known in the literature and will be provided. The goal is then to develop link the local stress state and a particular fluid velocity in a model which predicts the rate of sanding. Literature describing general criteria for sanding and associated volume of produced sand will also be provided.

In general, as the local stress exceeds certain level, a certain total amount of rock is fractured into sand. Then, depending on the flow rate this sand is removed at a certain rate. Finally, as sand is removed, the wellbore geometry changes. The main utility of a 1D model is to determine:

1. Sensitivity to global stresses and fluid pressure.
2. Derive an approximation for the change in wellbore radius as a result of sand removal.

Such a toy model will help our company in designing complex 3D local models. In such models, the wellbore trajectory and surface is resolved by a local grid. A good understanding of the radially symmetric response will be invaluable to generating complex 3D models with field-scale quantities coupled two-ways to the local response.

FINAL REPORTS

Compressed representation of a partially defined integer function over multiple arguments

Nina Daskalova, Plamen Mateev, Stela Zhelezova

1. The problem

In OLAP (OnLine Analitical Processing) data are analysed in an n-dimensional cube. The cube may be represented as a partially defined function over n arguments:

$$f(x_1, x_2, \ldots, x_n), \text{ where } x_i \in [1 \div N_i] \text{ for } i \in [1 \div n].$$

Considering that often the function f is not defined everywhere, is there a known way of representing f or the points, in which it is defined, in a more compact manner than the trivial one

$$x_1, x_2, \ldots, x_n, \quad f(x_1, x_2, \ldots, x_n)?$$

The goal is to reduce the time necessary for moving or storing f and using part of the time gained for computations for restoring the original f.

2. The team propositions

We consider the example data for this problem and observe that some argument combinations may be missing because the function is partially defined, but no one combination can repeat. So a combination of function arguments for some particular value is unique (Table1). Our idea is to replace the whole long list of function arguments with a single unique number (ID), which depends on this list. Then the function will be defined in the following way: (ID, f) instead of the trivial one.

The criteria to choose ID are:

- easy calculated from the argument list;
- uniqueness;
- easy to calculate back the argument list from the ID.

We offer two methods to assign ID to each different argument list.

ID	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
	1	2	1	5
	2	2	1	2
	2	2	2	7
	4	2	2	1
	4	3	1	6

Table 1: $f(x_1, x_2, x_3), x_1 \in [1 \div 4], x_2 \in [1 \div 3], x_3 \in [1 \div 2]$.

2.1. ID is a consecutive number of the corresponding argument list

An argument list is a n-tuple over some alphabet (from 1 to $max(N_1, \ldots, N_n)$) so we can order the different n-tuples lexicographically. The number of all possible combinations of function arguments is $N_1 N_2 \ \ldots \ N_n$. Then we use the consecutive number of the argument list in this order as ID.

For our example in Table 1 all possible triples are: $(1,1,1)$; $(1,1,2)$; $(1,2,1)$; $(1,2,2)$; $(1,3,1); \ldots (4,3,2)$. Then instead of presenting f as $(1,2,1,5)$ we use only the couple $(3,5)$ (the argument list $1,2,1$ is number 3 in the accepted lexicographic order).

There are two possibilities to find an ID, defined above.

First, if the firm works with a client many times, it is possible to keep at both applications (for the firm and for the client) all argument lists in lexicographic order and their corresponding numbers.

If the clients are quite different each time and argument lists also change dynamically, such storage is ineffective. In this case the ID number for each argument list can be obtained by:

$$ID = N_2 N_3 \ldots N_n (x_1 - 1) + N_3 N_4 \ldots N_n (x_2 - 1) + \\ \ldots + N_{n-1} N_n (x_{n-2} - 1) + N_n (x_{n-1} - 1) + x_n.$$

To reverse this formula the next procedure could be used:

```
x_n = ID mod N_n; ID = ID div N_n
for  k = (n-1) to 2 do
x_k = ID mod N_k + 1;  ID = ID div N_k
```

2.2. ID is a number in some numeral system (for example binary), different from the decimal one

Now we use $(1111001, 5)$ to present f instead of $(1,2,1,5)$, because $121_{10} = 1111001_2$. Remember that 1 byte = 8 bits and a number in decimal system is 1

byte while 8 positions in binary system are 1 byte. In our concrete example we obtain $3 \times 1 = 3$ byte vs. 1 byte.

Let $(*, *, \ldots, a_i, *, \ldots, *)$ denote an $(n-1)$-dimensional cube. The most frequently solved problem in OLAP is the computation of the sum of values F of a given sub-cube of the n-dimensional cube. In the case of the $(n-1)$-dimensional cube the values F are:

$F(*, *, \ldots, a_i, *, \ldots, *) =$

$\sum_{j_1}^{N_1} \sum_{j_2}^{N_2} \cdots \sum_{j_{i-1}}^{N_{i-1}} \sum_{j_{i+1}}^{N_{i+1}} \cdots \sum_{j_n}^{N_n} f(j_1, j_2, \ldots, j_{i-1}, a_i, j_{i+1}, \ldots, j_n)$

To compute the values of all sub-cubes

$$F(*, x_2, \ldots, x_n), \ldots, F(*, *, x_3, \ldots, x_n), \ldots, F(*, *, \ldots, *)$$

an extra symbol is added to the alphabet of each argument (Table 2). The argument lists are extended but the property, which we use to add an ID, is the same. Therefore our idea can be extended also on the new argument lists.

ID	x_1	x_2	x_3	$f(x_1, x_2, x_3)$
	1	2	1	5
	2	2	1	2
	2	2	2	7
	4	2	2	1
	4	3	1	6
	*	2	1	7
	*	2	2	8
	...	...	...	...
	4	*	*	7
	*	*	*	21

Table 2: $f(x_1, x_2, x_3), x_1 \in [1 \div 4], x_2 \in [1 \div 3], x_3 \in [1 \div 2]$

In addition our group suggests the firm to archive the final records trying another software.

The document compression standard DjVu [1] uses an algorithm called arithmetic coding. Arithmetic coding, invented by Jorma Rissanen, and turned into a practical method by Witten, Neal, and Cleary, achieves a superior compression. Free browser plug-ins and desktop viewers from different developers are available from the djvu.org website.

7z is a compressed archive file format with LZMA compression. Compression ratio results are very dependent upon the data used for the tests. Usually, 7-Zip compresses to 7z format 30-70% better than to zip format. And 7-Zip compresses to zip format 2-10% better than the most of other zip compatible programs [2].

References

[1] Léon Bottou, Patrick Haffner, Paul G. Howard, Patrice Simard, Yoshua Bengio and Yann Le Cun, High Quality Document Image Compression with DjVu, Journal of Electronic Imaging, 7(3): 410–425(1998).

[2] `http://www.7-zip.org/`

Estimation of errors in text and data processing

Angela Slavova, Borislav Valkov, Krasimir Tonchev, Nina Daskalova, Margarita Nikolova, Mira Bivas, Plamen Mateev, Roumyana Yordanova, Stela Zhelezova

1. Problem description

The company Adiss Lab Lts. obtained 1 000 000 medical reports that are either in free form text, or in XML format. One of the main goals of their development is to integrate an algorithm for information extraction (IE) in their platform. Since the algorithm will work with medical data, its performance should be as reliable as possible. Due to confidentiality, we were not given access to the data or to the algorithm, but only to a simple example of a report's content. That is why we considered them mostly as a black box. The verification of the algorithm's output for a report is done by a medical doctor (MD) for a certain fee. Validating the correctness of all data would be overwhelming and very expensive. Hence, the problem, as presented by the company, is to provide a method (algorithm) which determines the minimum amount of reports that will validate the correctness of the IE algorithm and a procedure for selecting these reports.

2. Considered approaches

In order to solve the problem we have considered two types of approaches:

- *algorithm centric* – active learning and semi-supervised learning;
- *data centric* – sampling methods (e.g. bootstrapping) and power calculation.

Since we do not have the data, we have settled on an algorithm centric approach, specifically the active learning.

3. Active learning

3.1. General algorithm

The primary goal of active learning (AL) [1] is exploiting unlabelled data. Nowadays such data is *plentiful and cheap* – e.g documents off the web, speech

samples, images and video. However labelling it is *expensive*, just as in our case. Here is the active learning typical algorithm:

1. Start with a pool of unlabeled data
2. Pick a few points at random and get their labels
3. Repeat:
   - Train a classifier using the labels seen so far
   - Label randomly selected unlabeled points that are closest to the decision boundary (or most uncertain)

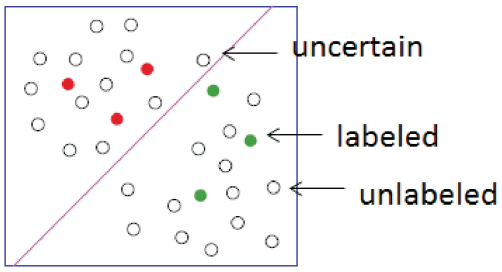


Figure 1: General AL algorithm

As it is seen in the algorithm, the active learning requires a *classifier with uncertainty output*. Another important specific is that *sampling is biased* i.e. the labelled samples are not representative for the underline distribution.

3.2. With clustering

An improvement of the active learning algorithm in our case would be to exploit the natural structure of the data by using *clustering*.
In general, clustering is incorporated in active learning [2] with the following steps:

1. Find a clustering of the data
2. Sample a few randomly-chosen points in each cluster and label them
3. If the granularity is right, assign each cluster its majority label; if not refine the clustering

Figure 2: Finding the right granularity of the clustering

4. Proposed solution of the problem

Our proposition is *active learning with clustering* to be used in order to select the minimal amount of reports required to validate the correctness of the information extraction. We propose an algorithm for doing this. Additionally we suggest the rules for clustering, measure of uncertainty for algorithm decision and measure of accuracy of the extraction.

4.1. Algorithm

The algorithm consists of the following steps:

1. Select a classifier suitable for active learning
2. Select a sampling scheme
3. Fix K – the number of samples labelled at each iteration
4. Select the stopping threshold $t > 0$
5. Extract the meta-data from each document (city, hospital, etc.)
6. Cluster the reports according to the meta-data
7. Select samples from each cluster using the sampling scheme, so that their total count equals to K
8. Label the selected samples and store them in a buffer
9. Do
   (a) Train the classifier with the data in the buffer
   (b) Apply the classifier to the remaining unlabelled data
   (c) Measure the accuracy of the current iteration (A_i) using all labelled samples

(d) Select the uncertain samples (close to the decision threshold)

(e) Select samples from the uncertain ones in each cluster using the sampling scheme, so that their total count equals to K

(f) Label the selected samples and store them in the buffer

While $|A_i - A_{i-1}| > t$

4.2. Algorithm details

Let us elaborate on the parts of the algorithm specific for the problem.

Clustering procedure. The clustering is based on the assumption that the MDs in same areas and hospitals have the same convention for writing reports. This leads to similarity in notations between the reports from the same area/ hospital/MD. We should note that this clustering may not be hierarchical.

Measure of uncertainty. Since we are dealing with medical data, we assume that the dictionary is fixed and known and can be reduced depending on the medical tests. Therefore, we treat the different phrases as elements of a vector. For each report the information extraction algorithm outputs for each phrase from the dictionary whether it is fully recognised, or its value is uncertain, or it is not tested.

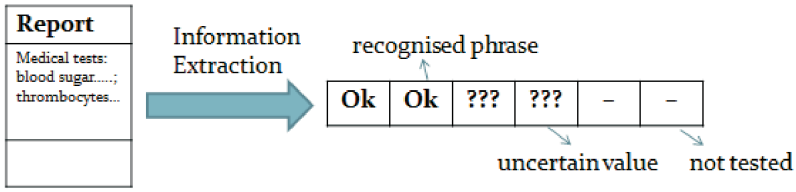


Figure 3: Output of the information extraction

We suggest the quotient of the count of the uncertain phrases - k divided by the count of all phrases - n to be used as a measure of uncertainty:

$$u = \frac{k}{n}, \quad u \in [0, 1]$$

Measure of accuracy. We suggest a few ways for measuring the accuracy of the algorithm.

The first one is the wide-known measures *precision* and *recall* [3] to be used. In order to define them, we introduce some terms. The terms true positives, true negatives, false positives, and false negatives compare the results of the

information extraction with the MD judgement. The terms positive and negative refer to the information extraction prediction, and the terms true and false refer to whether that prediction corresponds to the MD judgment. This is illustrated by the table below:

	actual class (observation)	
predicted class (expectation)	**tp** (true positive) Correct result	**fp** (false positive) Unexpected result
	fn (false negative) Missing result	**tn** (true negative) Correct absence of result

Precision and recall are then defined as:

$$Precision = \frac{tp}{tp + fp}$$
$$Recall = \frac{tp}{tp + fn}$$

Another suitable for the problem measure is one that combines precision and recall. This can be the harmonic mean of precision and recall, the traditional *F-measure* or balanced F-score [4]:

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The *Matthews correlation coefficient*(MCC) [4] can also be used for measuring the accuracy in our case. It is a measure of the quality of binary (two-class) classifications and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes. The MCC is in essence a correlation coefficient between the observed and predicted binary classification:

$$MCC = \frac{tp \cdot tn - fp \cdot fn}{\sqrt{(tp + fp)(tp + fn)(tn + fp)(tn + fn)}}$$

5. Suggestions for implementing our solution

Our solution can be easily implemented using open-source tools for active learning. We suggest the DUALIST (`https://code.google.com/p/dualist/`) interactive machine learning system to be used. Its main advantages are that it

quickly building classifiers for text processing tasks and also has web-based user interface in Java.
For further information on active learning, we recommend the *Active Learning, Synthesis Lectures on Artificial Intelligence and Machine Learning* book by Burr Settles (`http://active-learning.net/`).

References

[1] Dasgupta S. and Langford J., *A tutorial on active learning*, ICML 2009.

[2] Dasgupta S. and Hsu D., *Hierarchical sampling for active learning*, ICML 2008.

[3] Van Rijsbergen C. J., *Information Retrieval* (2nd ed.), Butterworth, 1979.

[4] Baldi P., Brunak S., Chauvin Y., Andersen C. A. F., Nielsen H., *Assessing the accuracy of prediction algorithms for classification: an overview*, Bioinformatics 2000, 16, p. 412-424

Optimization of the charging process in zinc hydrometallurgy

Alexander Peltekov, Doychin Boyadzhiev, Mariusz Kozlowski,
Zlatko Varbanov, Zlatogor Minchev

Problem definition

The main objective is to optimize the charging process in zinc metallurgy with respect to the technological price of the input components' mixture (concentrate build of different charges):

$$L(\vec{x}) = \sum_{j=1}^{n} C_{T_j} x_j \rightarrow \min, \tag{1}$$

where C_{T_j} is the technological price of the j-th concentrate and x_j – the quantity of the j-th concentrate in the charge (measured in [%]). Further on, for simplicity C_{T_j} is denoted without the j-th index, i.e. – "C_T".

Calculation of the technological price

After the examples from [1] the technological price C_T could be considered as a triple component:

$$C_T = C_1 - C_2 - C_3. \tag{2}$$

The C_1 component depends on the Zn (zinc) contents in the concentrate as follows:

$$C_1 = \text{if } [(1-p) * Zn < q \text{ then } (Zn - q) * C_D/100 \text{ else } p * Zn * C_D/100], \tag{3}$$

C_D – daily price of Zn on London Metal Exchange (LME);
p – final agreed Zn content at the official LME (85%);
q – minimum deduction of Fe (iron) at the official LME (8%).

The second component C_2 is the correction that depends on the difference between C_D and C_{avg} of Zn:

$$C_2 = E_{avg} - (C_{avg} - C_D)/10, \tag{4}$$

where E_{avg} are the average expenses for one tone concentrate processing and transportation, $E_{avg} = 2000$ \$.

The third component C_3 is the penalty for Fe contents in the concentrate:

$$\text{if } [Fe < q \text{ then } 0 \text{ else}(Fe - q) * s], \tag{5}$$

where q is the minimum deduction of Fe at the official LME (8%) and s – the penalty for 1 % Fe contents in the concentrate, $s = 2$ \$.

Methods and techniques

Assuming that the concentrate mixture is a linear combination of the charge components we decided to use the well-known linear optimization techniques including both – real and integer ones. As the input data example could be presented in a table-like form we decided to use the commonly available *MS EXCEL*©2010 product with its build-in SOLVER [2].

The input data arrays that were used contain percentages of the chemical elements, components and groups of a certain imaginary concentrate (see first column of Table 2) in twenty-one real Zn concentrates, denoted by "**ZC17-1"** – **"ZC17-21"** (see the first column of Table 1).

Results and future work perspectives

The first task that was solved concerns the creation of a concentrate mixture with "minimal price", keeping in mind the technological constraints for percentages contents of desired variables (noted in the second column of Table 2). The first two variables that have been monitored are Zn and S_{tot}. We have assumed the following rule: if the boundary conditions have a positive influence to the concentrate mixture, we use relations like "not less than", whilst the negative influence is denoted by relations like "not more than". The different variables share in the concentrates is limited by the percentages sum (that should be equal to 100%).

In order to achieve similarity in the utilized procedure and for experimental purposes (for other, different but similar optimization problems solving) we put the constraint for the technological price C_T of the desired concentrate contents

Table 1

Concentrates (in [%])	Charge 1 **Price 543.55**	Charge 2 **Price 549.61**	Charge 3 **Price 558.03**	Charge 4 **Price 560.00**	Charge 5 **Price 559.25**	Charge 6 **Price 558.03**	Charge 7 **Price 560.00**	Charge 8 **Price 557.86**	Charge 9 **Price 558.03**
ZC17-1	12.35		5			5	10.56		5
ZC17-2	12.26	12	15	11.53	15	15	11.95	14	15
ZC17-3								1	
ZC17-4	20.08	17	35	32.32	31	35	29.00	25	35
ZC17-5	7.91	3	5	0.99	1	5	5.25	2	5
ZC17-6	14.84	18	35	1.08	16	35	19.56	23	35
ZC17-7									
ZC17-8									
ZC17-9	6.53	3		16.64	1		8.09	1	
ZC17-10					3				
ZC17-11									
ZC17-12	18.91	17		0.25			9.86	11	
ZC17-13		5		2.45	5			5	
ZC17-14					4				
ZC17-15									
ZC17-16		15	5	30.05	15	5		10	5
ZC17-17		1			2			1	
ZC17-18	1.94	3		4.69	2		3.28	2	
ZC17-19		4							
ZC17-20	4.15	2			3		2.08	4	
ZC17-21	1.04				2		0.38	1	

(following the above rule, and in the present example: not more than 600\$/t, marked by $*$ at the last row of the second column of Table 2) keeping in mind the current Zn LME price of 1880\$/t. The LME is a parameter that automatically produces changes in the technological prices of the different concentrate components. We have found out that such mixture exists and this price boundary condition is not producing an unfeasible problem.

The obtained concentrate is of technological price $C_T = 543.55$ \$/t. The percentage distribution of the different components is given in column "Charge 1" of Table 1, and the chemical contents in column "Charge 1" of Table 2.

This solution is a theoretical one and is aiming minimal expected price, but it is not practically applicable because the concentrate building is based on an imperfect transport mechanism ("clamshell excavator") that could make a limited (by means of equal volume size) grabs from the different silos that store the charges. That is why it is practically impossible to build a concentrate with fractional components, e.g. 12.35%, from the first component, 12.26% from the second one, etc.

So, an integer optimization problem has to be solved for the different charge components producing the desired concentrate that have to be of integer type. The resulting solution is of technological price $C_T = 549.61$ \$/t (that is greater than the previous, fractional one, because of this new boundary condition). The quantity of the different charge components is given in column "Charge 2" of Table 1, and its chemical contents in "Charge 2" column of Table 2.

The resulting new solution is practically applicable if the concentrate is built with 100 grabs of the "clamshell excavator" (putting together each grab to one percentage from the concentrate).

This unfortunately is also not always possible taking into account the continuous production process.

So, we tried to find a solution that allows concentrate charges to be aliquot to five, producing in this way the desired ones with twenty grabs (which sounds more applicable in practice).

For this task solving we used twenty-one new, integer variables y_j, putting $x_j = 5y_j$ $(j = 1, 2, \ldots, 21)$. The different concentrates share in the resulting solution is given in column "Charge 3" of Table 1 and its chemical structure in column "Charge 3"of Table 2. The new calculated technological price is: $C_T = 558.03$ \$/t.

As it is clear from the solutions of these three problems that the resulting technological price C_T oscillates in-between 543 \$/t and 558 \$/t. We have to pay attention that the resulting difference is not quite big (less than 3% from the

Table 2

Components	*Bound*	**Minimal price**			*Bound*	**Minimal Pb + Cu + SiO_2**			*Bound*	**Minimal Fe**		
		Charge 1	Charge 2	Charge 3		Charge 4	Charge 5	Charge 6		Charge 7	Charge 8	Charge 9
Zn	**50**	52.584	52.901	53.305	**50**	53.610	53.486	53.305	**50**	53.522	53.350	53.305
S_{tot}	**31**	31.000	31.081	32.130	**31**	31.506	31.120	32.130	**31**	31.000	31.151	32.130
Pb	**2.5**	2.000	1.922	1.951	**2.5**	1.104	1.819	1.951	**2.5**	1.900	1.973	1.951
Cu	**1**	1.000	0.999	0.998	**1**	1.000	0.997	0.998	**1**	1.000	0.999	0.998
SiO_2	**2.3**	2.300	2.295	2.199	**2.3**	2.300	2.065	2.199	**2.3**	2.300	2.211	2.199
Fe	**8**	8.000	7.992	7.984	**8**	8.000	7.992	7.984	**8***	7.640	7.817	7.984
Se	**20**	6.335	6.270	7.450	**20**	6.953	6.870	7.450	**20**	6.930	6.670	7.450
Hg	**10**	2.392	2.310	2.650	**10**	2.775	2.930	2.650	**10**	2.567	2.600	2.650
Cd	**0.3**	0.254	0.226	0.230	**0.3**	0.196	0.221	0.230	**0.3**	0.249	0.230	0.230
Ni	**15.1**	15.100	14.300	13.500	**15.1**	15.100	14.700	13.500	**15.1**	15.100	14.600	13.500
Co	**100**	69.493	65.540	71.500	**100**	40.916	73.460	71.500	**100**	67.376	73.480	71.500
As	**0.08**	0.064	0.065	0.040	**0.08**	0.064	0.064	0.040	**0.08**	0.064	0.065	0.040
Sb	**160**	159.208	142.310	156.300	**160**	159.000	158.800	156.300	**160**	159.104	154.920	156.300
Ge	**30**	4.792	4.900	5.000	**30**	5.000	4.850	5.000	**30**	4.896	4.800	5.000
Te	**20**	10.685	9.720	8.350	**20**	7.486	7.660	8.350	**20**	9.337	9.110	8.350
Pb + Cu	**3**	3.000	2.920	2.949	**3**	2.104	2.816	2.949	**3**	2.900	2.971	2.949
Pb + Cu + SiO_2	**5.5**	5.300	5.215	5.147	**5.5***	4.404	4.881	5.147	**5.2#**	5.200	5.182	5.147
Sb + Ge	**164**	164.000	147.210	161.300	**164**	164.000	163.650	161.300	**164**	164.000	159.720	161.300
Sb + Ge + As	**808**	808.000	799.510	562.800	**808**	808.000	799.950	562.800	**808**	808.000	806.920	562.800
Mn	**0.45**	0.230	0.228	0.162	**0.45**	0.102	0.201	0.162	**0.45**	0.184	0.218	0.162
CaO	**0.8**	0.531	0.473	0.377	**0.8**	0.239	0.412	0.377	**0.8**	0.434	0.475	0.377
MgO	**0.3**	0.180	0.185	0.167	**0.3**	0.122	0.141	0.167	**0.3**	0.164	0.172	0.167
Al_2O_3	**0.5**	0.386	0.371	0.270	**0.5**	0.185	0.272	0.270	**0.5**	0.311	0.348	0.270
Cl	**0.08**	0.080	0.074	0.013	**0.08**	0.080	0.073	0.013	**0.08**	0.080	0.080	0.013
F	**0.02**	0.010	0.008	0.005	**0.02**	0.005	0.008	0.005	**0.02**	0.007	0.008	0.005
Tl	**0.005**	0.002	0.001	0.002	**0.005**	0.001	0.001	0.002	**0.005**	0.002	0.001	0.002
Price	**600***	543.554	549.608	558.027	**560#**	560.000	559.252	558.027	**560#**	560.000	557.863	558.027

price).

Thus, we posted another optimization goal: "To find concentrate with technological price $C_T \leq 560$ \$/t (the bound is marked by $*$ in Table 2), taking a new boundary condition: minimize $\sum Pb + Cu + SiO_2 \leq 5.5\%$ from the used concentrate mixture (this condition is marked by $*$ in Table 2).

The idea is to keep a reasonable C_T and at the same time to diminish the negative influence of the above mentioned chemical substances. This problem was chosen, because the contents of "$Pb + Cu + SiO_2$" has a negative influence on the concentrate further roasting process and thus should be minimized.

So, we have solved three new integer continuous optimization problems (similar to the already posted above, i.e. aliquot to five) with a new optimization function goal: to minimize the percentage of this components' sum in the concentrate.

The resulting concentrates shares in the different charges are given in columns "Charge 4", "Charge 5" and "Charge 6"of Table 1, and their chemical components – in columns "Charge 4", "Charge 5" and "Charge 6" of Table 2. This generates the percentages of the concentrate negative components to be optimized from 5.3 % to 4.4% in the real optimization case and from 5.2% to 4.9% in the integer one, keeping the same results when the boundary condition for grabs aliquot to five is posted.

Another chemical element that has a negative influence to the *Zn* production process is the *Fe* presence. So we posted the following: minimize the *Fe* content and keeping $C_T \leq 560$ \$/t (the bound is marked by # in Table 2) for $\sum Pb + Cu + SiO_2 \leq 5.2\%$ from the used concentrate mixture. The bounds are marked by $*$ (for "*Fe*") and by # (for "$Pb + Cu + SiO_2$") in Table 2.

Three more similar to the already described optimization problems have been solved and the resulting solutions are given in columns "Charge 7", "Charge 8" and "Charge 9" of Table 1 and Table 2.

A minimal *Fe* discount is noticed for the first two cases, whilst the third (for grabbing aliquot to five) produced no visible changes.

It was also noted that the fourth concentrate, ZC17-4, is used in all of the above mentioned results, but the eight one, ZC17-8, is not.

Further, we have solved three new optimization tasks, limiting the fourth concentrate from above (ZC17-4 $\leq 30\%$) and the eight concentrate from below (ZC17-8 $\geq 5\%$) in order to include it in the resulting concentrate. We do not give the solution details for these three problems. We mark only the results: the real optimization gives $C_T = 548.83$ \$/t, the integer optimization gives $C_T \leq 553.41$ \$/t, and the third one, that uses an integer optimization with grabbing aliquot to five, has not a feasible solution.

Conclusions

By using the proposed methodology the technologists can be oriented on the possible combination of the different concentrates. The consistent application of this methodology can provide an acceptable as chemical composition and technologically feasible in the real production process charge, which is optimal under well-defined criteria and constraints.

References

[1] Söderström, U. Zinc smelter revenue stream, Boliden's Capital Markets Days, Kokkola Smelter, Finland, November 4–5, 2008, Available at: http://investors.boliden.com/afw/files/press/boliden/Kokkola-2008-5_Smelters_Zinc_US.pdf

[2] Christopher Zappe, Ch., Winston W., Ch. Albright, Data Analysis /Optimization / Simulation Modelling with Microsoft Excel, International Edition, 2011, ISBN 10: 0538476125 / ISBN 13: 9780538476126

Mini Max Wallpaper

Antonio Marigonda, Dragomir Aleksov, Jan Idziak,
Krasimir Georgiev, Mariusz Kozlowski, Mikhail Krastanov,
Milena Veneva, Monika Sikora, Slav Angelov

Executive summary

Problem: Mini Max company formulated a problem for automatic calculation the number of wallpaper rolls necessary for decorating a room with wallpapers. The final goal is the development of a web-based calculator open for use to both Mini Max staff and the general public.

Proposed solution: We propose an approach for reducing the studied problem to the one-dimensional cutting-stock problem. We show this in details for the case of plain wallpapers as well as for the case of patterned wallpapers with straight match. The case of patterned wallpapers with offset match can be considered similarly.

The one-dimensional cutting-stock problem can be formulated as a linear integer programming problem. The approach proposed by P. C. Gilmore and R. E. Gomory in 1961 can be used for solving this problem for the case when the number of variables is large. In our opinion, the dynamic programming approach or the approach based on generating of all possible cuttings is more appropriate for the case of separate room for which a relatively small number of wallpapers' strips is needed.

We develop an approach for calculating the needed number of wallpapers for relatively small problems, create an algorithm in a suitable graphical interface and make different tests. The tests show the efficiency of the proposed approach compared with the existent (available) wallpapers' calculators.

1. Introduction

For 20 years now Mini Max Ltd company distributes in Bulgaria high quality wallcoverings manufactured by leading European companies. In its showroom can be found a wide variety of designs, original patterns and stylish combinations appropriate for both residential and commercial environments. A professional

team assists clients in selecting the right wallpapers. But there is a problem: how to calculate the necessarily amount of rolls.

Everyone can find different types of wallpapers' calculators on the Internet (cf., for example, the following WEB addresses

http://www.diy.com/diy/jsp/bq/templates/content$_1$ookup.jsp?content
= /content/knowledge/calculators/wallpaper/wallpaper.jsp
http://www.tangletree − interiors.co.uk/wallpaper − calculator/
http://www.praktiker.bg/praktiker − international/html/bg$_B$G/
161211/index.html
http://www.wallpaperdirect.com/wallpaper − calculator.php?)

requiring usually the sizes of the walls (and, in some cases, the sizes of a door and/or windows).

Simple tests convinced us that these calculators are not appropriate for the practical needs of clients. And this is our main motivation for studying this problem.

2. Problem statement

Now we shall state the problem in full details.

2.1. Data of the problem

The client will provide:

- number of walls to be decorated;
- width and height of each wall;
- sizes of door and windows present in each wall (if any).

The wallpaper rolls are characterized by:

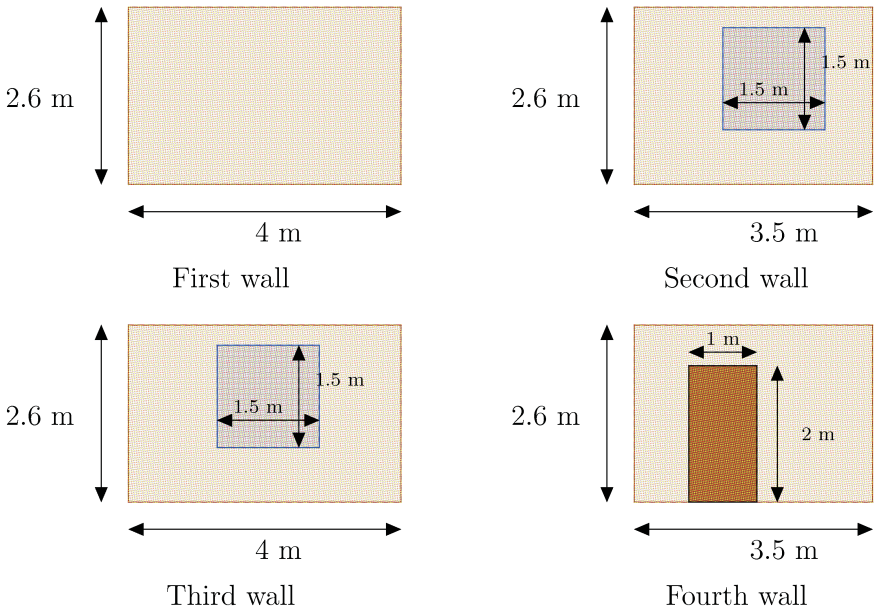
- width and length of the roll;
- design of the roll (plain, straight pattern, offset pattern).
- for each roll with straight pattern, it is specified the height of the pattern.
- for each roll with offset pattern, it is specified the height of the pattern and the offset.

2.2. Restrictions

($R1$) On each wall can be used only one kind of roll.
($R2$) Each wall should be fully covered with vertical stripes, without overlapping.
($R3$) It is forbidden to join two stripes along horizontal in horizontal line.
($R4$) In the case of straight pattern, two consecutive stripes must match their pattern.
($R5$) In the case of offset pattern, the patterns between two consecutive stripes must have an offset specified by the offset parameter of the roll.

3. Basic example

We assume to have to decorate a standard room with two windows and one door, using rolls of length 10 m, width 0.7 m, and straight pattern of height 0.64 m as in the following basic example:



4. Room analysis

Now we will begin our analysis of the room. Assume that the room has d walls, we start considering a matrix $\tilde{W}$ whose i-th row contain the width w_i and

the height h_i of the i-th wall of the room.

$$\tilde{W} := \begin{pmatrix} w_1 & h_1 \\ \vdots & \vdots \\ w_d & h_d \end{pmatrix}.$$

In the basic example, the matrix is:

$$\tilde{W} := \begin{pmatrix} 4 & 2.6 \\ 3.5 & 2.6 \\ 4 & 2.6 \\ 3.5 & 2.6 \end{pmatrix}.$$

We have to take into account the following further technical restriction:

(T_1) When cutting the wallpaper to size for the required wall height, it is necessary to take into account a 10 cm safety margin, which is used in the case the ceiling or floor is not completely straight.

So we will consider all the walls as 10 cm longer than in the given data, forming the new matrix:

$$W := \begin{pmatrix} w_1 & h_1 + 0.1 \\ \vdots & \vdots \\ w_d & h_d + 0.1 \end{pmatrix},$$

in our example:

$$W := \begin{pmatrix} 4 & 2.7 \\ 3.5 & 2.7 \\ 4 & 2.7 \\ 3.5 & 2.7 \end{pmatrix}.$$

4.1. Plain wall

We will call *plain walls* the walls of the room in which there are neither windows, nor doors. The situation is the one depicted in the First Wall of the basic example. This is the most simple situation.

In the straight pattern case, since we have to match the pattern along consecutive stripes, we will impose that all the strips should begin with the beginning of a pattern.

The number of patterns required on each strip with patterns of height h_P in order to fill the height of the i-th wall is:

$$\ell^i_{\max} := \left\lceil \frac{h_i + 0.1}{h_P} \right\rceil,$$

where we set:

$$\lceil x \rceil = \min\{n \in \mathbb{N} : \, x \leq n\},$$
$$\lfloor x \rfloor = \max\{n \in \mathbb{N} : \, x \geq n\}.$$

In our basic example, we have that $h_P = 0.64$, $h_i = 2.6$, $i = 1, \ldots, 4$, whence

$$\ell^i_{\max} := \left\lceil \frac{2.7}{0.64} \right\rceil = 5, \qquad i = 1, \ldots, 4.$$

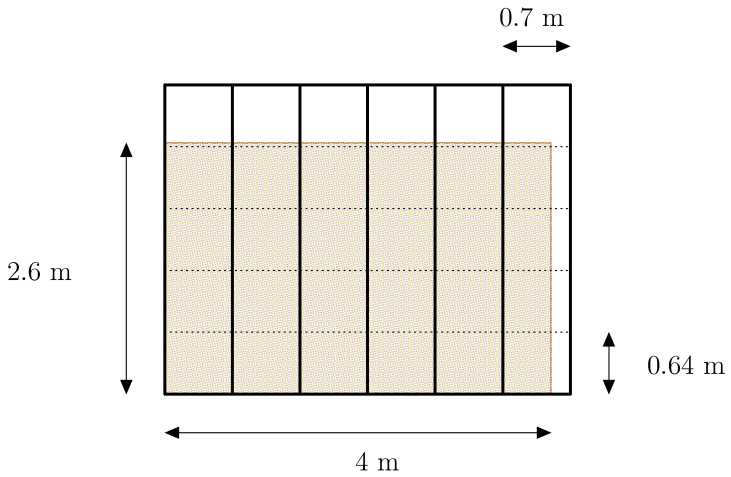
The number of strips of width w_P needed to fill the i-th wall is given by

$$n_i := \left\lceil \frac{w_i}{w_P} \right\rceil.$$

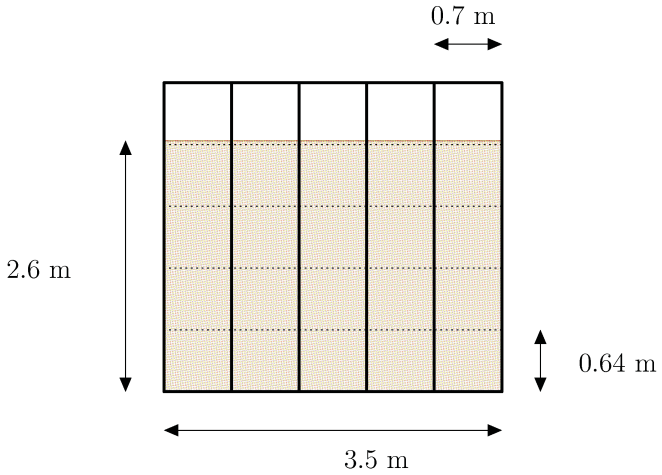
In this computation it **does not matter** if the wall contains windows or door.

In our basic example, we have $w_P = 0.7$ m, whence:

$$n_1 = n_3 = \left\lceil \frac{4}{0.7} \right\rceil = 6,$$
$$n_2 = n_4 = \left\lceil \frac{3.5}{0.7} \right\rceil = 5.$$



The First and Third wall of the basic example:
6 strips containing 5 patterns are needed.

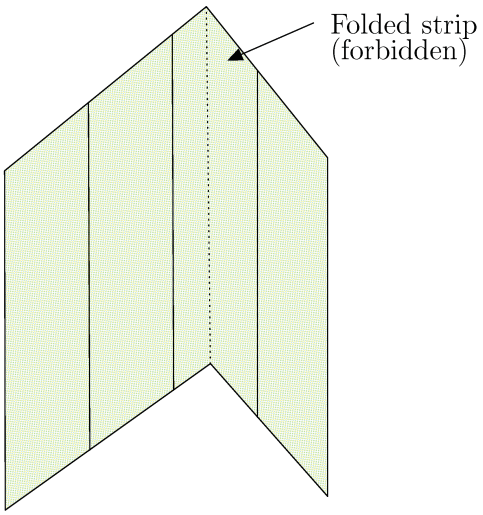


The Second and Fourth wall of the basic example:
5 strips containing 5 patterns are needed.

The part exceeding the measure of the wall is considered *wasted*. In particular we are assuming that the following condition holds:

(G_1) The part of the strip that exceeds the width of the wall cannot be used to another wall.

In other words, (G_1) forbids us to *fold* a strip around a corner.



Assumption (G_1) forbids to fold strips around corners.

If we *do not* assume that (G_1) holds, i.e. we *allow* folding, the problem is reduced to the situation in which we have a *single* wall.

4.2. Managing obstacles

If a window or a door is present on a wall, then we say that there is an *obstacle*. If the obstacle is sufficiently large, we may spare a certain amount of wallpaper. However, since the exact knowledge of the position of obstacles in the wall is unknown (we have information only about the size of each obstacle), we will consider the worst case scenario, i.e. we imagine that the obstacles are always placed in the position that minimize the spare of wallpaper.

All the information on obstacles is encoded in the following matrix:

$$\mathcal{O} := \begin{pmatrix} o^1_{wl} & o^1_w & o^1_h \\ \vdots & & \vdots \\ o^k_{wl} & o^k_w & o^k_h \end{pmatrix},$$

where o^j_{wl} denotes the index of the wall containing the j-th obstacle, and o^k_w, o^k_h are its width and height, respectively. k is the total number of obstacles.

In our basic example, we have that:

$$\mathcal{O} := \begin{pmatrix} 2 & 1.5 & 1.5 \\ 3 & 1.5 & 1.5 \\ 4 & 1 & 2 \end{pmatrix}.$$

Consider now a generic obstacle of width o_w and o_h on a wall. We want to compute the minimum number of strips that will be fully horizontally cut by the presence of the obstacle. We will call them the strips *blocked* by the obstacle.

The maximum number of strips that can be entirely contained in the obstacle are $\left\lfloor \frac{o_w}{w_P} \right\rfloor$. However, if we perform a slight horizontal translation to the right, we have that the first strip now is not fully horizontal cut, and if the translation was sufficiently small, we did not cut in full the last one in which the translated obstacle is present. So the number of strips blocked by an obstacle of width o_w is

$$b(o_w) = \max\left\{0, \left\lfloor \frac{o_w}{w_P} \right\rfloor - 1\right\}.$$

The strips that encounter an obstacle, but are not blocked by it in the above sense, will be considered as the one not encountering the obstacle *at all*. Accordingly, every obstacle whose width is strictly less that $2w_P$ can be actually *neglected*.

In our example, the number of strips blocked by each window is

$$b(1.5) = \max\left\{0, \left\lfloor \frac{1.5}{0.7} \right\rfloor - 1\right\} = 1,$$

while the door is negligible since $b(1) = 0$.

The same argument is used to find the minimum number of patterns fully contained in an obstacle: indeed if a pattern of the strip is completely contained into an obstacle, we may cut the strip in order to *save* it. The worst case possible is the one in which the number of patterns that can be saved is minimal.

So the number of patterns that can be saved due to presence of an obstacle of height o_h is

$$s(o_h) = \max\left\{0, \left\lfloor \frac{o_h}{h_p} \right\rfloor - 1\right\}.$$

The stripes that are blocked by a thin obstacle that does not allow any sparing of patterns, will be considered as the one not encountering the obstacle *at all*. Accordingly, every obstacle whose height is strictly less that $2h_p$ can be actually *neglected*.

In our example, we can spare from each window

$$s(1.5) = \max\left\{0, \left\lfloor \frac{1.5}{0.64} \right\rfloor - 1\right\} = 1$$

pattern.

If in a wall there are more than one obstacle (i.e. many windows, or doors), we make the following assumption (*non interacting obstacles*):

(G_2) The sets of strips blocked by each obstacle are pairwise disjoint.

If assumption (G_2) is not fulfilled, we start to ignore some obstacles until we fulfill (G_2) with the remaining ones.

Under assumption (G_2), we are able for each wall to construct a list containing the number of strips and the number of pattern for each strips allowing us to cover it. Assume to have to cover the wall indexed by i, then we consider the rows of the matrix $\mathcal{O}$ whose first element is i:

$$J_i = \{j : \text{ the first element of the } j\text{-th row of } \mathcal{O} \text{ is } i\}.$$

If $j \in J_i$, j-th obstacle will allow us to spare $s(o_h^j)$ patterns for each of the $b(o_w^j)$ stripes that blocks. So in that wall we can use $b(o_w^j)$ stripes with $\ell^i_{\max} - s(o_h^j)$ patterns to cover the area where the obstacle is present. We stress on the fact

that this is the worst case scenario, indeed if the obstacle is well-placed, the spare of wallpaper can be much greater.

We partition J_i in the following way. For every $0 \leq k \leq \ell^i_{\max} - 1$ we have the (possibly empty) sets:

$$J_i^k := \{j \in J_i : \ell^i_{\max} - s(o_h^j) = k\},$$

Given $0 \leq k \leq \ell^i_{\max} - 1$, we will define

$$n_{i,k} = \begin{cases} \displaystyle\sum_{j \in J_i^k} b(o_w^j), & \text{if } J_i^k \neq \emptyset, \\ \\ 0, & \text{if } J_i^k = \emptyset, \end{cases}$$

$$n_{i,\ell^i_{\max}} = n_i - \sum_{k=0}^{\ell^i_{\max}-1} n_{i,k},$$

$$n_{i,m} = 0, \text{ if } p > \ell^i_{\max}.$$

Finally, we have that to cover the i-th wall we need $n_{i,k}$ stripes of length k, $k = 1, \ldots, \ell^i_{\max}$. Summing up on the indexes of the walls, we end up with the following matrix:

$$\mathcal{N} := \begin{pmatrix} \displaystyle\sum_{i=1}^{d} n_{i,1} & 1 \\ \vdots & \vdots \\ \displaystyle\sum_{i=1}^{d} n_{i,\ell_\infty} & \ell_\infty \end{pmatrix},$$

where

$$\ell_\infty := \max_{i=1,\ldots,d} \ell^i_{\max}.$$

This matrix encodes the total number of strips for each length that are needed in order to cover the room.

In our basic example, we have $\ell_\infty = 5$, $J_1 = J_4 = \emptyset$, $J_2 = J_2^1 = \{1\}$, $J_3 = J_3^1 = \{2\}$, $n_{1,5} = 6$, $n_{2,4} = n_{3,4} = 1$, $n_{2,5} = 4$, $n_{3,5} = 5$, $n_{4,5} = 5$, in all the other

cases we have $n_{p,q} = 0$. So

$$\mathcal{N} := \begin{pmatrix} 0 & 1 \\ 0 & 2 \\ 0 & 3 \\ 2 & 4 \\ 20 & 5 \end{pmatrix},$$

which means that to cover the walls of the room we need 2 stripes of length 4 and 20 stripes of length 5.

5. The one-dimensional cutting-stock problem

The one-dimensional cutting-stock problem can be formulated as follows: We assume that an unlimited number of wallpapers' rolls of length L is given. We need a given number b_i of pieces of length l_i with $l_i \leq L$ for $i = 1, 2, \ldots, m$. We say that the vector $(a_1, a_2, \ldots, a_m)$ whose components are nonnegative integers is an admissible cut if the following inequality holds true:

$$a_1 l_1 + a_2 l_2 + \cdots + a_m l_m \leq L.$$

Let us assume that n is the maximum number of all admissible cuts

$$a_j = (a_{1j}, a_{2j}, \ldots, a_{mj})^T,\ j = 1, 2, \ldots, n,$$

where by a^T is denoted the transposed vector of the vector a. If we denote by x_j, $j = 1, 2, \ldots, n$, the number of rolls needed to be cut using the admissible cut a_j, our problem consists in minimizing the sum

$$x_1 + x_2 + \cdots + x_n$$

in such a way that the needed amount of pieces of strips of wallpapers with fixed length to be satisfied.

In this way we can formalize the one-dimensional cutting-stock problem by considering the following linear integer programming problem (P):

$$x_1 + x_2 + \cdots + x_n \ \rightarrow\ \min$$

$$\begin{aligned} &a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1 \\ &a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2 \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ &a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m \end{aligned}$$

x_j is a nonnegative integer
for each index $j = 1, \ldots, n$.

5.1. An approach proposed by P. C. Gilmore and R. E. Gomory

To solve the problem (P) we can apply the approach proposed by P. C. Gilmore and R. E. Gomory in [2] and [3] (cf. also, for example, [1] and [4]). In our case, this approach consists in solving the linear problem (P_0)

$$x_1 + x_2 + \cdots + x_n \ \rightarrow \ \min$$

$$\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{array}$$

$$x_j \geq 0, \ j = 1, \ldots, n.$$

Without loss of generality we may write this problem in the following form

$$< e, y > + \sum_{i \in N} z_i \ \rightarrow \ \min$$

$$\begin{array}{c} By + \sum\limits_{j \in N} a_j z_j = b \\ y_j \geq 0, \ z_k \geq 0, \end{array}$$

where B is the basic matrix, e is a vector of length m with components equal to 1 and $< \cdot, \cdot >$ denotes the standard scalar product in R^m. When we start to solve this problem we set

$$B = \begin{pmatrix} \lfloor L/l_1 \rfloor & 0 & 0 & \ldots & 0 \\ 0 & \lfloor L/l_2 \rfloor & 0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & \lfloor L/l_m \rfloor \end{pmatrix}.$$

Let us underline that the columns a_j, $j \in N$, are unknown. Then from the presentation

$$y = B^{-1}b - \sum_{j \in N} B^{-1}a_j z_j.$$

we can replace each basic variable in the target function and to obtain that the coefficient in front of the j-th nonbasic variable x_j in the target function is $c_j = 1 - e^T B^{-1} a_j$, $j \in N$. If all coefficients c_j, $j \in N$, are nonnegative, then this will be the solution of the problem (P_0).

But the columns a_j are in fact unknown. In order to understand if there exists an admissible cut a for which the coefficient $1-e^T B^{-1}a < 0$, the following integer knapsack problem (K_0) is considered:

$$p_1 a_1 + p_2 a_2 + \cdots + p_m a_m \to \max$$

$$l_1 a_1 + l_2 a_2 + \cdots + l_m a_m \le L$$

a_j is a nonnegative integer
for each index $j = 1, \ldots, m$.

where $p^T = e^T B^{-1}$.

Let $\bar{a}$ denotes its solution. If $p^T \bar{a} > 1$, then $\bar{a}$ is admissible and the cut should be added as a new column to the matrix of the considered linear problem (P_0) and to repeat the same procedure. If $p^T \bar{a} \le 1$, we are sure that there is no admissible cut that has to be taken into account for solving the problem (P_0).

5.2. "A brute force algorithm"

Data of the algorithm: We have a matrix $\mathcal{N}_{m\times 2}$, where the first element of i-th row N_{i1} denotes the needed number of strips containing the number of patterns specified by the second element N_{i2}.

Steps of the algorithm

Step 1: We calculate the vector $w = (w_1, w_2, \ldots, w_m)$ with

$$w_i = \min\left(\left\lfloor \frac{L}{l_i} \right\rfloor, N_{i1}\right)$$

which estimates the number of strips that is reasonable to be obtained from a roll of wallpapers. Here L denotes the number of patterns contained in a roll and l_i, $i = 1, \ldots, m$ - the number of patterns contained in each needed strip.

Step 2: We form a set of all possible different vectors $v = (v_1, v_2, \ldots, v_m)$ such that each component v_i belongs to the set $\{0, 1, \ldots, w_i\}$. Each element of this set represents an abstract cut of a roll. Then we form the matrix A whose columns are the elements of this set.

Step 3: From the matrix A we form the matrix B by eliminating the zero vector as well as all the columns $v = (v_1, v_2, \ldots, v_m)$ of the matrix A satisfying the inequality

$$v_1 w_1 + v_1 w_1 + \cdots + v_m w_m > L.$$

The columns of the matrix B represent all admissible cuts of a roll.

Step 4: From the matrix B we form the matrix C by eliminating all the columns $v = (v_1, v_2, \ldots, v_m)$ of the matrix B satisfying the following: $m-1$ components of the vector v are zeros and there exists another column $\bar{v}$ of B also with $m-1$ zeros on the same places as v such that the nonzero component of v is strictly less that the same component of $\bar{v}$. The matrix C is a subset of admissible cuts of a roll that are more efficient than the columns of B, i.e. if we can fulfill some needs with the cuts from B we can do the same needs with the cuts from C.

Step 5: We form the augmented matrix

$$D = [B : C : \cdots : C]$$

where the number of copies of the matrix C is equal to

$$M = \max\left\{\left\lceil \frac{N_{i1}}{\lfloor L/l_i \rfloor} \right\rceil\right\} - 1.$$

The number $M+1$ gives us an upper bound of the needed number of cuts to realize each kind of strip number separately. So the total cuts needed to realize all requirements is bounded from above by the number of columns of the matrix B $+ M *$ (number of columns of C). We have to pay your attention that replacing the matrix B with C we can not ensure to fulfill the needed requirements as equalities.

Step 6: So we formulate the linear algebraic system $Dx = b$ with b equal to the first column of the matrix $\mathcal{N}$, where the components of the unknown vector x take values only 0 and 1. We are minimizing the sum of the components of the vector x by using a standard procedure from Matlab. The minimum of the sum of the components gives us the minimum rolls needed to fulfill the requirements. The columns corresponding to the nonzero components of x determine the optimal way of cuts of each roll.

5.3. The recursive solution

Our starting point is the two-column matrix $\mathcal{N}$ encoding the number of strips needed for each length, together with the number of patterns in a roll L. Assume that $\mathcal{N}$ has m rows. We can reduce $\mathcal{N}$ to the case in which the first element of each row is nonzero. Define E_i to be the $m \times 2$ matrix with 1 in the $(m, 1)$ entry and 0 elsewhere.

We define now the following recursive function
$\mathrm{Roll} : \mathrm{Mat}_{m,2}(\mathbb{N}) \times \{1, \ldots, m\}$ by setting:

- $\mathrm{Roll}(M, p) = 1$ if the first column of M is the zero vector
- If this is not the case, we form the m-dimensional vector $c = (c_1, \ldots, c_m)$ as follows:

$$c_j = \begin{cases} \mathrm{Roll}(M - E_j, p - M_{j2}), & \text{if } M_{i1} > 0,\ p \geq M_{j2} \\ \\ +\infty \text{ otherwise,} \end{cases}$$

and set

$$\mathrm{Roll}(M, p) = \begin{cases} 1 + \mathrm{Roll}(M, L), & \text{if } c_j = +\infty \text{ for every } j = 1, \ldots, m, \\ \\ \min\limits_{j=1,\ldots,m} c_j, & \text{otherwise.} \end{cases}$$

This recursion procedure can be improved also keeping track of the decision made at each step, hence providing not only the optimal number of rolls, but also the optimal way of cutting. Unfortunately, the time needed to have a solution with this procedure is considerably sensitive with respect to the size of the matrix $\mathcal{N}$ and also to the size of the entries.

References

[1] A. C. Dikili, B. Barlas, *A Generalized Approach to the Solution of One-Dimensional Stock-Cutting Problem for small Shipiyards*, Journal of Marine Science and Technology, Vol. 19, No. 4, pp. 368–376, 2011.

[2] P. C. Gilmore and R. E. Gomory, *A Linear Programming Approach to the Cutting-Stock Problem*, Operations Research, Vol. 9, No. 6 (Nov.–Dec., 1961), pp. 849–859, 1961.

[3] P. C. Gilmore and R. E. Gomory, *A Linear Programming Approach to the Cutting-Stock Problem – Part II*, Operations Research, Vol. 11, No. 6 (Nov.–Dec., 1963), pp. 863–888, 1963.

[4] R. W. Haessler, P. E. Sweeny, *Cutting-Stock Problems and Solution procedures*, Europenian Journal for Operations Research, Vol. 54, pp. 141–150, 1991.

Laboratory calibration of a MEMS accelerometer sensor

Irina Georgieva, Clemens Hofreither, Teodora Ilieva,
Tihomir Ivanov, Svetoslav Nakov

1. Introduction

Nowadays, systems, using microelectromechanical systems (MEMS) sensors (e.g. accelerometers, gyroscopes, etc.) find more and more applications. For instance, they are used for video stabilization, mobile applications and many other things. Among the reasons for their wide use is the fact that they are very cheap and have miniature size (usually less than $10 \times 10 \times 5$ mm) and consume little power. However, there are a lot of unexplored fields (like navigation) where they could be used once the necessary mathematical knowledge is developed. For further information, see for example [4].

In the present work we address MEMS accelerometers in particular. There are many sources of error that affect their behaviour [4]. In this report, we present two mathematical models and associated algorithms for calibrating the device, i.e. compensating for the constant bias error (the offset of its output signal, when the device does not undergo any acceleration), scaling errors (the deviation of the accelerometer's scale from the unit we want to use, like m/s^2) and nonorthogonality of the axis.

2. Approach 1

This approach was studied by Teodora Ilieva, Tihomir Ivanov and Svetoslav Nakov.

We have a system, consisting of three accelerometers, that measure the linear acceleration in three "almost" orthogonal directions. We shall assume that the angle between each two of those directions is $90^\circ \pm 2\%$ [5]. The relationship between accelerometer's outputs and the measured acceleration can be well modeled as a linear function [1]. In Subsection 2 we present the calibration algorithm. Similar approach to the problem is proposed, e.g., in [2], [3] (see also the references within those papers). There a six-parameter and a nine-parameter linear

models are developed for calibrating the device. Both of those approaches use an orthogonal coordinate system, with respect to which the corresponding analysis is carried. We study the problem by working in the coordinate system defined by the directions in which the accelerometers measure linear acceleration. We find this to be the more natural way and, thus, simplifying the analysis. Also, an usual thing to do is to approximate the sine and the cosine functions of small angles with the respective angles (which leads to the linearity of the aforementioned models). We do not use this approximation in order to obtain maximal accuracy in the model.

In Subsection 3 we study computationally the error that can be caused by not considering the non-orthogonality of the axis. We show that even small deviations in the directions of the accelerometers can lead to catastrophic error in the estimated position of the body within a few seconds of integrating.

2.1. Mathematical model describing the relationship between the measured and the real values of the acceleration

Let $\hat{a_x}$, $\hat{a_y}$ and $\hat{a_z}$ be the real values that the accelerometers show and a_x, a_y and a_z be the corresponding calibrated values of the acceleration in the x, y and z directions, respectively, with $\vec{e_1}$, $\vec{e_2}$ and $\vec{e_3}$ being unit vectors in those directions. We use the following model to calibrate the accelerometers [1]:

$$\begin{aligned} a_x &= (\hat{a_x} - s_1)/b_1 \\ a_y &= (\hat{a_y} - s_2)/b_2, \\ a_z &= (\hat{a_z} - s_3)/b_3 \end{aligned} \tag{1}$$

where s_1, s_2, s_3 (which we call shifts) and b_1, b_2, b_3 (which we call scaling coefficients) are yet to be determined. In matrix form (1) can be written as

$$\underbrace{\begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}}_{\boldsymbol{a}} = \underbrace{\begin{bmatrix} 1/b_1 & 0 & 0 \\ 0 & 1/b_2 & 0 \\ 0 & 0 & 1/b_3 \end{bmatrix}}_{T} \underbrace{\begin{bmatrix} \widehat{a_x} \\ \widehat{a_y} \\ \widehat{a_z} \end{bmatrix}}_{\widehat{\boldsymbol{a}}} - \underbrace{\begin{bmatrix} s_1/b_1 \\ s_2/b_2 \\ s_3/b_3 \end{bmatrix}}_{\boldsymbol{s}}.$$

Using the introduced notation, we obtain the equation

$$\boldsymbol{a} = T\widehat{\boldsymbol{a}} - \boldsymbol{s} \tag{2}$$

Proposition 1. *If we assume that the axis of the accelerometers are orthogonal, then the acceleration is*

$$a_{orth} := \sqrt{a_x^2 + a_y^2 + a_z^2}$$

Proof. The proof is obvious, using the fact that the sum of the squares of the directional cosines is 1. □

Since the axis are not orthogonal, let us denote the angles between the axis as follows (see Fig.1):

$$\phi := \angle(Ox, Oy),\ \psi := \angle(Ox, Oz),\ \theta := \angle(Oy, Oz) \tag{3}$$

Let $\vec{a}$ be an arbitrary acceleration acting on the sensor. Let $\overline{a_x}$, $\overline{a_y}$ and $\overline{a_z}$ be the affine projections of $\vec{a}$ onto the x, y and z axis, respectively. It is important to note that the measured values[1] of the acceleration in the x, y and z directions are not the affine projections of $\vec{a}$ onto the axis. Let us denote (see Fig. 1)

$$\alpha := \angle(\vec{a}, \vec{e_1}), \beta := \angle(\vec{a}, \vec{e_2}), \gamma := \angle(\vec{a}, \vec{e_3})$$

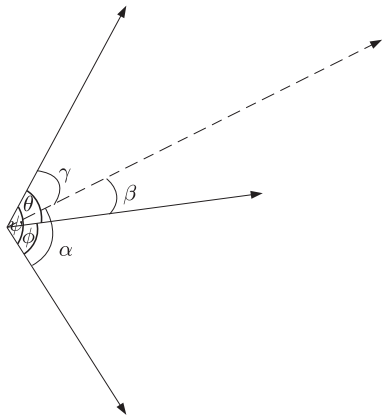


Figure 1: Acceleration vector in the affine coordinate system

It can be easily seen that the following Lemma holds true:

Lemma 1. *The equations*

$$\begin{aligned} a_x &= \|\vec{a}\| \cos\alpha \\ a_y &= \|\vec{a}\| \cos\beta \\ a_z &= \|\vec{a}\| \cos\gamma \end{aligned} \tag{4}$$

are valid.

[1]For the time being, let us think that the accelerometeres measure correctly, i.e. a_x, a_y and a_z are the measured values from the accelerometers.

We want to express the affine projections $\overline{a_x}$, $\overline{a_y}$ and $\overline{a_z}$ of $\vec{a}$ onto the coordinate axis with a_x, a_y and a_z.

Lemma 2. *For the affine projections of the acceleration $\overline{a_x}$, $\overline{a_y}$ and $\overline{a_z}$ the following equalities hold true:*

$$
\begin{aligned}
\overline{a_x} &= \frac{-\sin^2\theta a_x + (\cos\phi - \cos\theta\cos\psi)a_y + (\cos\psi - \cos\phi\cos\theta)a_z}{-1 + \cos^2\phi + \cos^2\psi + \cos^2\theta - 2\cos\phi\cos\psi\cos\theta} \\
\overline{a_y} &= \frac{(\cos\phi - \cos\theta\cos\psi)a_x - \sin^2\psi a_y + (\cos\theta - \cos\phi\cos\psi)a_z}{-1 + \cos^2\phi + \cos^2\psi + \cos^2\theta - 2\cos\phi\cos\psi\cos\theta} \\
\overline{a_z} &= \frac{(\cos\psi - \cos\phi\cos\theta)a_x + (\cos\theta - \cos\phi\cos\psi)a_y - \sin^2\phi a_z}{-1 + \cos^2\phi + \cos^2\psi + \cos^2\theta - 2\cos\phi\cos\psi\cos\theta}
\end{aligned} \tag{5}
$$

Proof. We have

$$
\vec{a} = \overline{a_x}\vec{e_1} + \overline{a_y}\vec{e_2} + \overline{a_z}\vec{e_3}, \tag{6}
$$

where $\vec{e_1}$, $\vec{e_2}$ and $\vec{e_3}$ are the unit vectors in the x, y and z directions, respectively. Consecutively, we take the scalar products of the both sides of (6) with $\vec{e_1}$, $\vec{e_2}$, $\vec{e_3}$ and obtain:

$$
\begin{aligned}
(\vec{a}, \vec{e_1}) &= \overline{a_x} + \overline{a_y}\cos\phi + \overline{a_z}\cos\psi \\
(\vec{a}, \vec{e_2}) &= \overline{a_x}\cos\phi + \overline{a_y} + \overline{a_z}\cos\theta \\
(\vec{a}, \vec{e_3}) &= \overline{a_x}\cos\psi + \overline{a_y}\cos\theta + \overline{a_z}
\end{aligned} \tag{7}
$$

Taking into account (4) we obtain

$$
\begin{bmatrix} 1 & \cos\phi & \cos\psi \\ \cos\phi & 1 & \cos\theta \\ \cos\psi & \cos\theta & 1 \end{bmatrix} \begin{bmatrix} \overline{a_x} \\ \overline{a_y} \\ \overline{a_z} \end{bmatrix} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix} \tag{8}
$$

We solve the system (8) and obtain (5). □

Remark 1. Let us note that the system (8) has diagonally dominant matrix and, therefore, it has a unique solution.

Remark 2. In matrix form, equations (5) can be rewritten as

$$
\overline{\boldsymbol{a}} = \overline{T}\boldsymbol{a},
$$

where:

$$\overline{\boldsymbol{a}} = \begin{bmatrix} \overline{a_x} \\ \overline{a_y} \\ \overline{a_z} \end{bmatrix},$$

$$\overline{T} = \frac{1}{den} \begin{bmatrix} -\sin^2\theta & \cos\phi - \cos\theta\cos\psi & \cos\psi - \cos\phi\cos\theta \\ \cos\phi - \cos\theta\cos\psi & -\sin^2\psi & \cos\theta - \cos\phi\cos\psi \\ \cos\psi - \cos\phi\cos\theta & \cos\theta - \cos\phi\cos\psi & -\sin^2\phi \end{bmatrix},$$

$$den = -1 + \cos^2\phi + \cos^2\psi + \cos^2\theta - 2\cos\phi\cos\psi\cos\theta,\ \boldsymbol{a} = \begin{bmatrix} a_x \\ a_y \\ a_z \end{bmatrix}$$

Lemma 2. *For the acceleration $a_{nonorth}$ the following holds true*

$$a_{nonorth} = \sqrt{\overline{a_x}^2 + \overline{a_y}^2 + \overline{a_z}^2 + 2\overline{a_x a_y}\cos\phi + 2\overline{a_x a_z}\cos\psi + 2\overline{a_y a_z}\cos\theta}. \tag{9}$$

Proof. Taking $a = a_{nonorth} = (\overline{a_x}, \overline{a_y}, \overline{a_z})$ in (6) and taking the scalar squares of both sides, we obtain

$$(a_{nonorth}, a_{nonorth}) = \overline{a_x}^2 + \overline{a_y}^2 + \overline{a_z}^2 + 2\overline{a_x a_y}(\vec{e_1}, \vec{e_2}) + 2\overline{a_x a_z}(\vec{e_1}, \vec{e_3}) + 2\overline{a_y a_z}(\vec{e_2}, \vec{e_3}),$$

where $\vec{e_1}$, $\vec{e_2}$, $\vec{e_3}$ are the unit vectors along the axis. Taking into account (3), (9) follows directly. □

We substitute (5) into (9) and simplify to obtain the following:

Proposition 2. *The equality*

$$a_{nonorth} = \sqrt{\frac{numerator}{denominator}}, \tag{10}$$

where

$$\begin{aligned} numerator =& 2(-1 + 1\cos 2\theta)a_x^2 + 2(-1 + 1\cos 2\psi)a_y^2 + 2(-1 + 1\cos 2\phi)a_z^2 \\ &+ 8(\cos\phi - \cos\psi\cos\theta)a_x a_y + 8(\cos\theta - \cos\phi\cos\psi)a_y a_z \\ &+ 8(\cos\psi - \cos\phi\cos\theta)a_x a_z; \\ denominator =& 2 + 2\cos 2\phi + 2\cos 2\psi + 2\cos 2\theta - 8\cos\phi\cos\psi\cos\theta \end{aligned}$$

holds true.

Now we shall derive an explicit expression for the calibrated values of the acceleration in an orthonormal cooridnate system, given the data, measured by the sensors.

Proposition 3. *Let $\vec{e_1}$, $\vec{e_2}$, $\vec{e_3}$ be the unit vectors in the directions, defined by the accelerometers' axis. Let us introduce an orthonormal coordinate system $O_{\vec{f_1},\vec{f_2},\vec{f_3}}$, where $\vec{f_1} \equiv \vec{e_1}$, $\vec{f_2}$ is in the $\vec{e_1}\vec{e_2}$ plane. Then the following equality is valid:*

$$\begin{bmatrix} \vec{e_1} \\ \vec{e_2} \\ \vec{e_3} \end{bmatrix} = \overline{T}^{orth} \begin{bmatrix} \vec{f_1} \\ \vec{f_2} \\ \vec{f_3} \end{bmatrix}, \tag{11}$$

where

$$\overline{T}^{orth} = \begin{bmatrix} 1 & \cos\phi & \cos\psi \\ 0 & \sin\phi & \dfrac{\cos\theta - \cos\phi\cos\psi}{\sin\phi} \\ 0 & 0 & \dfrac{\sqrt{1-\cos^2\phi-\cos^2\psi-\cos^2\theta+2\cos\phi\cos\psi\cos\theta}}{\sin\phi} \end{bmatrix} \tag{12}$$

Proof. Let us denote (see Fig. 2)

$$\xi := \angle(\vec{e_3}, \vec{f_2}),\ \eta := \angle(\vec{e_3}, \vec{f_3}).$$

Since $\vec{f_2}$ lies in the $\vec{e_1}\vec{e_2}$ plane and $\vec{f_1} \equiv \vec{e_1}$ we have the relationship

$$\vec{e_2} = b_1\vec{f_1} + b_2\vec{f_2}, \tag{13}$$

where b_1 and b_2 are real numbers. By taking the inner product of (13) successively with $\vec{f_1}$ and $\vec{f_2}$ we obtain

$$b_1 = \cos\phi,\ b_2 = \cos|90^\circ - \phi| = \sin\phi$$

Therefore, we have

$$\vec{e_2} = \cos\phi\vec{f_1} + \sin\phi\vec{f_2}$$

and, thus,

$$\vec{f_2} = -\frac{\cos\phi}{\sin\phi}\vec{e_1} + \frac{1}{\sin\phi}\vec{e_2}. \tag{14}$$

Analogously, we obtain

$$\vec{e_3} = \cos\psi\vec{f_1} + \cos\xi\vec{f_2} + \cos\eta\vec{f_3},$$

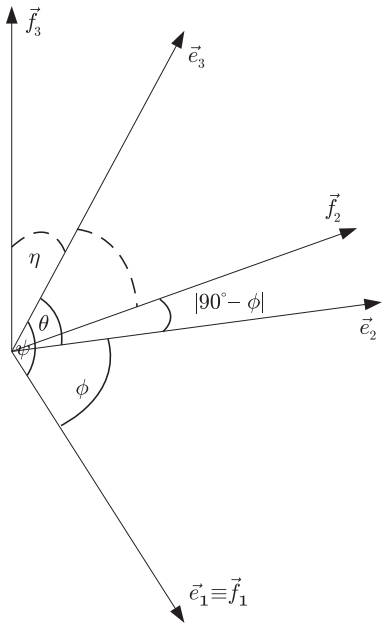


Figure 2: Orthonormal coordinate system

By taking into account (14), we obtain

$$\begin{aligned}\cos\xi = \left\langle \vec{e_3}, \vec{f_2} \right\rangle &= \left\langle \vec{e_3}, -\frac{\cos\phi}{\sin\phi}\vec{e_1} + \frac{1}{\sin\phi}\vec{e_2} \right\rangle \\ &= -\frac{\cos\phi}{\sin\phi}\cos\psi + \frac{\cos\theta}{\sin\phi} = \frac{\cos\theta - \cos\phi\cos\psi}{\sin\phi}.\end{aligned}$$

Furthermore, since $\vec{e_3}$ is a vector in the orthonormal coordinate system $O_{\vec{f_1}\vec{f_2}\vec{f_3}}$ and the angles with the axis being ψ, ξ and η, respectively the following holds true:

$$\begin{aligned}&\cos^2\eta + \cos^2\xi + \cos^2\psi = 1 \\ &\cos\eta = \sqrt{1 - \cos^2\xi + \cos^2\psi} \\ &= \frac{\sqrt{1 - \cos^2\phi - \cos^2\psi - \cos^2\theta + 2\cos\phi\cos\psi\cos\theta}}{\sin\phi}\end{aligned}$$

Let us note, that we have used the fact that η is small angle and, therefore, $\cos\eta > 0$. □

Corollary 1. *Let us have an arbitrary acceleration $\vec{a}$ with coordinates in the affine coordinate system $O_{\vec{e_1}\vec{e_2}\vec{e_3}}$, defined by the accelerometers' axis, $\overline{a_x}$, $\overline{a_y}$, $\overline{a_z}$,*

respectively. Let the corresponding coordinates in the $O_{\vec{f}_1\vec{f}_2\vec{f}_3}$ *coordinate system (defined as in Proposition* 3*) be* $\overline{a_x}^{orth}$, $\overline{a_y}^{orth}$ *and* $\overline{a_z}^{orth}$. *Then we have*

$$\begin{bmatrix} \overline{a_x}^{orth} \\ \overline{a_y}^{orth} \\ \overline{a_z}^{orth} \end{bmatrix} = \overline{T}^{orth} \begin{bmatrix} \overline{a_x} \\ \overline{a_y} \\ \overline{a_z} \end{bmatrix}, \tag{15}$$

where $\overline{T}^{orth}$ *is defined as* (12).

Let us denote

$$\overline{\boldsymbol{a}}^{orth} := \begin{bmatrix} \overline{a_x}^{orth} \\ \overline{a_y}^{orth} \\ \overline{a_z}^{orth} \end{bmatrix}, \ \overline{\boldsymbol{a}} := \begin{bmatrix} \overline{a_x} \\ \overline{a_y} \\ \overline{a_z} \end{bmatrix}.$$

Furthermore, let is take into account (2), Remark 2 and Corollary 1 and the notation, introduced in those. Then we obtain the following important result:

Theorem 1. *Let* $\widehat{a_x}$, $\widehat{a_y}$ *and* $\widehat{a_z}$ *be the values shown by the accelerometers at a certain time and* $\overline{a_x}^{orth}$, $\overline{a_y}^{orth}$ *and* $\overline{a_z}^{orth}$ *be the coordinates of the corresponding acceleration in the orthonormal coordinate system, defined in Proposition* 3. *Then the following holds true:*

$$\overline{\boldsymbol{a}}^{orth} = \overline{T}^{orth}\overline{T}T\widehat{\boldsymbol{a}} - \overline{T}^{orth}\overline{T}\boldsymbol{s}. \tag{16}$$

2.2. The calibration algorithm

Let us first note that when the device does not undergo any acceleration, then the length of the vector that the acceleraometers should show is equal to the acceleration due to the gravitational force. The reason for this fact is that the accelerometers do not measure acceleration directly, but force. Now we are ready to explain the algorithm itself.

1. First, we make measurements with the three accelerometer sensors in 20 different positions, when the device does not undergo any acceleration. Let us denote those measurements with $\hat{a}^{(i)}$, where $i = \overline{1, 20}$.

2. We make a "rough" estimate of the parameters in order to evaluate their signs. We achieve this by considering the simplified model, assuming the axis are orthogonal. Therefore, we find the values that minimize the sum

$$\sum_{i=1}^{20} \left[(T\hat{a}^{(i)} - s)^T (T\hat{a}^{(i)} - s) - g^2 \right]^2$$

3. Using a least squares fit, we find the parameters in the model that minimize the sum

$$\sum_{i=1}^{20}\left[(\overline{T}^{orth}\overline{T}T\hat{a}^{(i)}-\overline{T}^{orth}\overline{T}s)^T(\overline{T}^{orth}\overline{T}T\hat{a}^{(i)}-\overline{T}^{orth}\overline{T}s)-g^2\right]^2, \tag{17}$$

under the restrictions, imposed for the signs of the parameters by Step 2.

Let us note that Step 2 is important, because otherwise the numerical methods that we use for minimizing (17) do not always converge to the right values for the estimated parameters.

2.3. Estimates for the error due to nonorthogonality

We want to evaluate what is the error that can be caused by not considering the nonorthogonality of the axis. Let us denote

$$err(a_x,a_y,a_z,\phi,\psi,\theta):=|a_{nonorth}-a_{orth}|.$$

Since in the calibration procedure we use the fact that when the accelerometers are at rest, then the only acceleration that they measure is due to gravity, we consider the following extremal problem:

Problem 1.

$$\begin{aligned} &err(a_x,a_y,a_z,\phi,\psi,\theta)\longrightarrow \max,\\ &a_{nonorth}=g,\\ &0.98\pi\le\phi,\psi,\theta\le 1.02\pi, \end{aligned} \tag{18}$$

where g is the gravitational acceleration.

Let us note that the restrictions for ϕ, ψ, θ are imposed because of the up to 2% nonorthogonality of the axis.

By solving this problem numerically, we obtain the estimate of the maximal error

$$\max err(a_x,a_y,a_z,\phi,\psi,\theta)\approx 0.3130299\ m/s^2$$

This value is reached, for example at

$$a_x=a_y=a_z\approx 5.48114,\phi=\psi=\theta\approx 1.60221$$

Next, we want to give an estimate for the upper bound of the relative error that can be reached when the accelerometer measures an arbitrary acceleration. We assume consecutively that in each direction the accelerometer can measure

acceleration up to $\pm 4g$, $\pm 8g$, $\pm 16g$. In all the cases the estimate is approximately the same.

Problem 2.

$$\begin{aligned} &\left| \frac{a_{nonorth} - a_{orth}}{a_{nonorth}} \right| \longrightarrow \max, \\ &-16g \leq a_x, a_y, a_z \leq 16g, \\ &0.98\pi \leq \phi, \psi, \theta \leq 1.02\pi, \end{aligned} \tag{19}$$

The numerical solution of this extremal problem is approximately 3.192%.

Next, we present a histogram which represents the distribution of the relative error for values of ϕ, ψ, θ, respectively 1.53938, 1.60221, 1.60221, which are the estimated with the calibration procedure values for the sensor used in the tests (see Fig. 3).

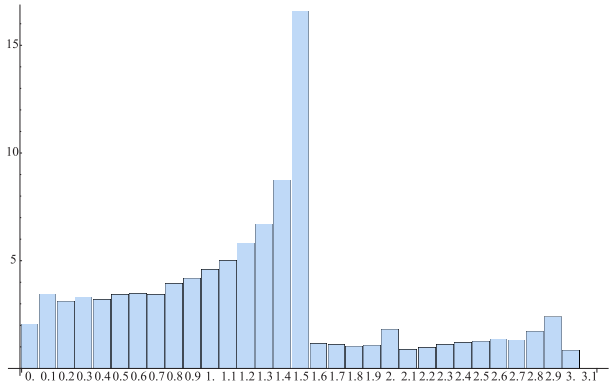
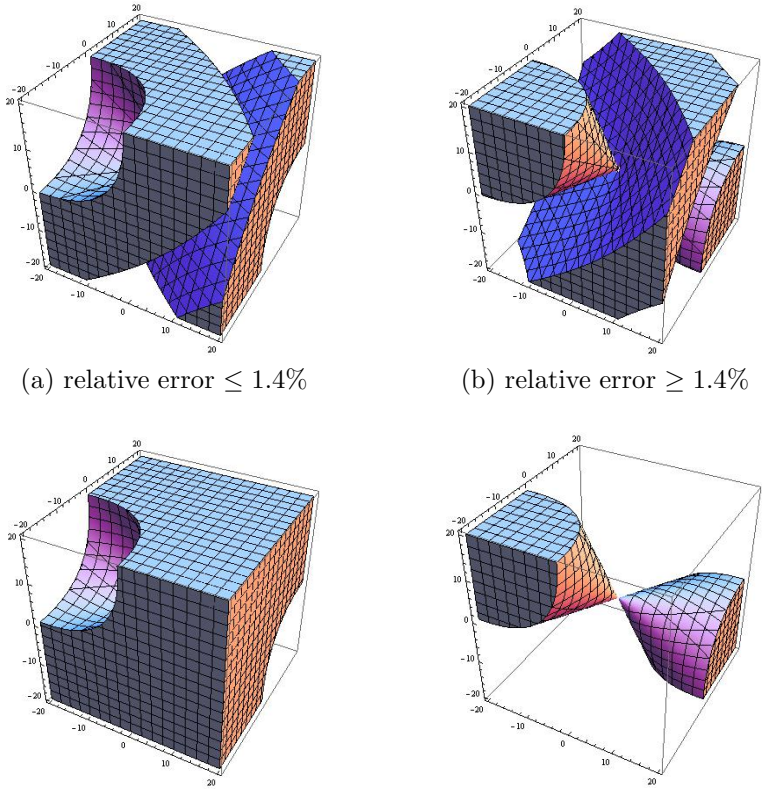


Figure 3: Distribution of the relative error

In order to be able to understand the "jump" in the histogram (Fig. 3), we present geometrically the domains in $\mathbb{R}^3$ where the relative error is respectively $\leq 1.4\%$, $\geq 1.4\%$,, $\leq 1.6\%$, $\geq 1.6\%$ (see Fig.4). We assume again that ϕ, ψ, θ are, respectively 1.53938, 1.60221, 1.60221, and a_x, a_y, a_z lie in the interval $[-20, 20]$.

All of those results show that the error that is due to the non-orthogonality of the axis is very serious one and cannot be neglected in order to obtain a linear model for the acceleration.



(a) relative error $\leq 1.4\%$ (b) relative error $\geq 1.4\%$

(c) relative error $\leq 1.6\%$ (d) relative error $\geq 1.6\%$

Figure 4: Domains in $\mathbb{R}^3$ for the relative error

2.4. Conducted experiments

In this subsection we present some experiments, that we have conducted in order to test the algorithm that we propose. We have tested it with 3 datasets from 3 different phones, each consisting of averaged measurements of the sensors in 27 different positions when the device does not undergo any acceleration. We have used 20 of those positions for calibrating the device and the other 7 we have used to check the results, by comparing the already calibrated acceleration with g.

- **Dataset 1.** In Table 1 the measured data from the three accelerometer sensors is shown.

  – First, we calibrate the device, using positions 1–20, i.e. we find the

Position	x	y	z
1	−9.54983	0.37829	−0.999283
2	10.145	0.314528	−1.37789
3	0.413309	−9.47641	−1.62076
4	0.368766	10.1442	−0.910232
5	0.510292	0.338026	−10.7851
6	0.231834	0.482186	8.60553
7	0.230219	7.85729	5.09836
8	0.304598	8.0554	−7.09352
9	−0.612011	−7.07391	−7.39908
10	−0.741095	−7.0657	5.22667
11	8.34517	−1.00171	4.35023
12	7.90356	−1.01121	−7.15935
13	−7.21644	−0.678396	5.06995
14	−7.53767	−0.887223	−6.8451
15	0.359915	1.36831	8.5543
16	0.0692607	1.56694	−10.7058
17	0.491484	−0.827214	8.55852
18	0.0425242	−0.802927	−10.712
19	−1.00027	0.562196	−10.6961
20	1.765	0.425372	8.51681
21	−1.90224	7.05	5.60731
22	2.7946	7.32034	−7.46149
23	−6.09005	−5.89288	2.99988
24	−6.12614	−2.40266	−7.91477
25	−5.17908	−5.64109	4.39876
26	−6.40025	7.4125	0.119649
27	−2.75267	7.19283	5.10852

Table 1: Dataset 1

values of the parameters, that minimize the sum

$$\sum_{i=1}^{20} \left[(\overline{T}^{orth}\overline{T}T\hat{a}^{(i)} - \overline{T}^{orth}\overline{T}s)^T (\overline{T}^{orth}\overline{T}T\hat{a}^{(i)} - \overline{T}^{orth}\overline{T}s) - g^2 \right]^2,$$

where $\hat{a}^{(i)}$ is the vector, consisting of the measurements in the i-th position in Table 1. We obtain the following values for the calibration

parameters:

$$s_1 \to 0.304496, s_2 \to 0.321482, s_3 \to -1.08995,$$
$$b_1 \to -1.00457, b_2 \to -1.00147, b_3 \to -0.989292,$$
$$\phi \to 1.57646, \psi \to 1.57096, \theta \to 1.57301$$

– Analogously, using positions 3–23, we obtain:

$$s_1 \to 0.318321, s_2 \to 0.322794, s_3 \to -1.09059,$$
$$b_1 \to -1.00472, b_2 \to -1.00124, b_3 \to -0.989255,$$
$$\phi \to 1.56932, \psi \to 1.5709, \theta \to 1.57296$$

Using the obtained values for the calibration parameters, we evaluate the relative error over all measured positions, i.e. we estimate

$$\left| \frac{a-g}{g} \right|,$$

where a is the absolute value of the acceleration, calculated with formula (16). In all cases it is less than 0.21%.

- **Dataset 2.** In Table 2 the measured data from the three accelerometer sensors from the second device is shown.

  – Using positions 1–20, we obtain:

$$s_1 \to -0.469784, s_2 \to 0.608425, s_3 \to 0.208206,$$
$$b_1 \to -0.999573, b_2 \to -0.989443, b_3 \to -1.01176,$$
$$\phi \to 1.56219, \psi \to 1.56969, \theta \to 1.57057$$

  – Using positions 3–23, the estimated values are:

$$s_1 \to -0.466405, s_2 \to 0.616422, s_3 \to 0.207149,$$
$$b_1 \to -0.999212, b_2 \to -0.989644, b_3 \to -1.01181,$$
$$\phi \to 1.55782, \psi \to 1.56827, \theta \to 1.57215$$

Once again, the values are very close to one another. The worst relative error that is reached with those calibrations is below 0.45%.

Position	x	y	z
1	7.324190906	1.250997715	−5.785632422
2	−8.334818875	1.423071223	6.075933722
3	−0.55309323	−1.556750779	−9.465058152
4	0.3963949	−2.401782608	9.590842176
5	4.468067668	0.438564599	8.815483348
6	−0.25604359	−8.251388776	4.231704803
7	−10.19639516	−0.540933147	−0.043947822
8	9.243301882	−0.489659084	0.177376713
9	5.184601909	0.464227916	−7.888789547
10	−7.133046458	0.486715023	7.492167273
11	−3.544315793	−5.885572603	−6.514445229
12	1.927551054	−5.721122639	7.317886686
13	0.712132815	0.611587605	10.05427574
14	−0.516080232	−1.053660856	−9.563502952
15	−0.482513916	−0.519767116	10.04926657
16	−0.366261687	1.69466758	−9.659420825
17	−1.770079301	0.62190275	10.02684107
18	0.714671068	0.553163819	−9.626586644
19	−0.557896244	1.798565701	10.05206135
20	−0.633628967	7.907637143	6.736185309
21	−0.244376238	7.911662259	−6.344980284
22	−0.391871362	−6.71082086	−6.278043744
23	−1.109805182	−6.966091847	6.382627512
24	−10.2685151	0.6944625	0.553426001
25	9.263098671	0.708257972	1.16414443
26	−0.341591276	−9.051813894	0.526102398
27	−0.766906823	10.32812321	0.883673299
28	0.139849901	0.372395084	−9.689413968
29	−0.825407808	0.440694427	10.10000846

Table 2: Dataset 2

- **Dataset 3.** In the last dataset, shown in Table 3, it is important to note, that the measurements from positions 3 and 21 are very similar to each other. As one can see from the results shown below, if we use both those position for calibration purposes the result is much worse.

  - Using positions 1–20 (i.e. using only one of the similar rows of data), we obtain a relative error of 0.26%
  - Using positions 3–23, we obtain a relative error of 0.98%

  This shows that it is important to make the measurements in distinct po-

Position	x	y	z
1	0.686143985	9.693013241	0.146230973
2	0.307313184	−9.555131822	0.121707371
3	10.20588166	0.146627372	0.293913142
4	−9.235730337	0.149835656	−0.153514714
5	0.277144172	−0.108499369	9.72381854
6	0.652333397	0.026782191	−9.867581581
7	2.819218128	0.00984892	9.453752615
8	−1.006468208	0.586054944	9.610065897
9	10.10396889	0.089937928	−1.625049867
10	−1.605962585	0.058272144	−9.639987664
11	0.246794337	1.126126555	9.709396155
12	0.706211903	−3.139230967	−9.302827738
13	7.728566894	0.248629672	6.471431237
14	2.053861069	−2.927665847	9.11283197
15	5.742237957	0.041531765	8.152279472
16	1.604557037	0.951821853	−9.754449668
17	1.029262184	1.996550095	9.543562547
18	0.041407842	−2.516368282	9.358991009
19	2.730162016	−8.137597633	−4.711984149
20	−2.924918942	−6.411393964	6.335544161
21	10.18267071	0.144573621	0.312617999
22	3.908533423	−8.83618239	−1.558956159
23	−4.59854197	−8.151722848	0.065404421
24	−1.674747469	9.466063897	0.036135226
25	10.04636598	1.799748205	0.246702889
26	7.685992297	6.529638446	0.083514447

Table 3: Dataset 3

sitions. Otherwise, the least-squares fit is biased towards one position and the result gets worse.

2.5. Conclusion

We have derived an explicit formula for the linear acceleration, given the values of the accelerometers and the angles between the axis of the sensors. We have explained how one can find an approximation of those angles and the shifts and scaling coefficients, given some measurements of the sensors, when the system does not undergo any acceleration. We have run some numerical experiments in order to understand better the error, caused by the nonorthogonality of the axis.

Future work on the topic could include studying the error with which the calibration parameters are evaluated with the procedure that we propose. Also,

the optimal positions in which the calibration data is to be collected, should be determined. Probably, more investigation on the errors can give some insights about those positions.

3. Approach 2

This approach was studied by Irina Georgieva and Clemens Hofreither.

3.1. Mathematical problem description

We are concerned with an end-user device (e.g., smart phone) with a built-in accelerometer which measures accelerations along three axes. In the following, we refer to the true acceleration which acts on the device by a, and to the measurements which its sensor outputs by m.

In practice, the measurements m will not coincide with the true physical acceleration a due to many factors, among these,

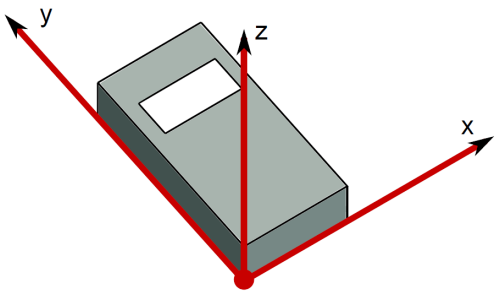
- scaling errors,
- crosstalk,
- offset errors,
- non-orthogonality of the accelerometer axes,
- time-dependent noise,
- etc.

We want to find and calibrate a model which allows us to reconstruct as well as possible the true accelerations from the measured data,

$$m \mapsto a,$$

by taking into account the systematic errors in the accelerometer of one particular device.

For all further considerations, we fix a coordinate system relative to the phone as shown in the below picture.



All accelerations a

$$a = (a_x, a_y, a_z)$$

acting on the phone will be considered in this coordinate system. We point out that this coordinate system may not be aligned with the axes of the accelerometer, and there is no need for this.

3.2. Mathematical model

We assume an affine-linear relationship between the measurements $m = (m_1, m_2, m_3)$ and the true acceleration vector $a = (a_1, a_2, a_3)$, that is, the function $m = F(a)$ should be of the form

$$m = F(a) = Xa + y.$$

Here, $X \in \mathbb{R}^{3\times 3}$ is an unknown matrix, and $y \in \mathbb{R}^3$ is an unknown vector. In this model, the diagonal of X may represent the scaling error, the off-diagonal entries of X the crosstalk factors, and y the offset. However, X can also represent arbitrary rotations in the case that the accelerometer sensor is not installed in an axis-aligned way in the device.

Once we have computed X and y, we can simply invert the above formula to obtain the true acceleration a from measurements m by the formula

$$a = F^{-1}(m) = X^{-1}(m - y). \tag{20}$$

This model has 12 parameters, namely, the 9 entries of X plus the 3 entries of y.

3.3. Determining the parameters

Let us assume that we have put the device into a standard orientation for which the actual forces a which act on it are known. For instance, putting the device facing up on a flat table, we can assume that the force is

$$a = (0, 0, -\mathrm{g}),$$

where $g \approx 9.81\,\mathrm{m/s^2}$ is the gravitational acceleration. By reading the output of the sensor and averaging over a sufficiently long span of time in order to reduce the measurement noise, we obtain a measured acceleration $m = (m_1, m_2, m_3)$.

By our model, we know that

$$Xa + y = m,$$

or in component form,

$$\begin{aligned} x_{11}a_1 + x_{12}a_2 + x_{13}a_3 + y_1 &= m_1, \\ x_{21}a_1 + x_{22}a_2 + x_{23}a_3 + y_2 &= m_2, \\ x_{31}a_1 + x_{32}a_2 + x_{33}a_3 + y_3 &= m_3. \end{aligned}$$

Remember that X and y are unknown, and we collect all their components into a single vector $z \in \mathbb{R}^{12}$ of unknowns,

$$z = (x_{11}, x_{12}, x_{13}, x_{21}, x_{22}, x_{23}, x_{31}, x_{32}, x_{33}, y_1, y_2, y_3)^\top, \tag{21}$$

for which we want to solve. Rewriting the above three equations in matrix-vector form, we obtain

$$\begin{pmatrix} a_1 & a_2 & a_3 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & a_1 & a_2 & a_3 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & a_1 & a_2 & a_3 & 0 & 0 & 1 \end{pmatrix} z = m.$$

Denoting the matrix in the above equation by $K(a) \in \mathbb{R}^{3\times 12}$, we can shortly write

$$K(a)\, z = m.$$

However, this system of linear equations has 12 unknowns, but only 3 equations and is therefore underdetermined.

In order to get a solution, we need to make a series of N measurements m for different known accelerations a. We will denote them by

$$\begin{aligned} a^{(1)} = (a_1^{(1)}, a_2^{(1)}, a_3^{(1)}), &\qquad m^{(1)} = (m_1^{(1)}, m_2^{(1)}, m_3^{(1)}), \\ \vdots &\qquad \vdots \\ a^{(N)} = (a_1^{(N)}, a_2^{(N)}, a_3^{(N)}), &\qquad m^{(N)} = (m_1^{(N)}, m_2^{(N)}, m_3^{(N)}). \end{aligned}$$

Setting up the linear system we have derived above for each of these measurements, we get a complete linear system of the form

$$\begin{bmatrix} K(a^{(1)}) \\ K(a^{(2)}) \\ \vdots \\ K(a^{(N)}) \end{bmatrix} z = \begin{bmatrix} m^{(1)} \\ m^{(2)} \\ \vdots \\ m^{(N)} \end{bmatrix}.$$

We call the block matrix on the left-hand side K and observe that it has size $3N \times 12$, and call $\overline{m}$ the right-hand side vector of size $3N$. In short, we have the linear system

$$Kz = \overline{m}.$$

If we take $N = 4$ measurements, the system is square, and it is uniquely solvable provided that the four vectors

$$\left(a_1^{(i)}, a_2^{(i)}, a_3^{(i)}, 1\right), \qquad i = 1, 2, 3, 4 \tag{22}$$

are linearly independent.

In practice, we recommend taking more measurements, say, at least $N = 6$ by measuring all six canonical orientations of a box-shaped device. The resulting linear system is then overdetermined. By using the standard technique of least-squares fitting, we can pass to the square linear system

$$K^\top K z = K^\top \overline{m}$$

which gives us the unique minimizer of

$$\left|\overline{m} - Kz\right|^2.$$

Again, this system has a unique solution provided that any four vectors from $\{a^{(i)}\}$ satisfy the linear independence condition from (22).

The matrix $K^\top K$ has size 12×12 and can thus easily and quickly be inverted by standard linear solvers, for instance, Gaussian elimination.

Once we have solved this linear system and obtained z, we can refer to formula (21) and simply extract the coefficients to obtain the original matrix X and the vector y. Then, from formula (20), we get the relation which allows us to compute the true accelerations a from arbitrary measurements m.

3.3.1. Simplification of the least-squares system

In practice, it is a very natural choice to have the known accelerations point along the positive and negative coordinate axes, i.e.,

$$\begin{aligned} a^{(1)} &= (-\mathrm{g}, 0, 0), \quad & a^{(3)} &= (0, -\mathrm{g}, 0), \quad & a^{(5)} &= (0, 0, -\mathrm{g}), \\ a^{(2)} &= (\mathrm{g}, 0, 0), & a^{(4)} &= (0, \mathrm{g}, 0), & a^{(6)} &= (0, 0, \mathrm{g}). \end{aligned}$$

In Subsection 3.4, we show how to obtain these measurements in a simple way. For this natural choice, it is easy to see that the least-squares matrix takes the form of a diagonal matrix, namely,

$$K^\top K = \mathrm{diag}(2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 2\mathrm{g}^2, 6, 6, 6).$$

Therefore, in this case, the least-squares problem

$$K^\top K z = K^\top \overline{m}$$

admits a simple solution in closed form,

$$X = \frac{1}{2\mathrm{g}} \begin{pmatrix} -m_1^{(1)} + m_1^{(2)} & -m_1^{(3)} + m_1^{(4)} & -m_1^{(5)} + m_1^{(6)} \\ -m_2^{(1)} + m_2^{(2)} & -m_2^{(3)} + m_2^{(4)} & -m_2^{(5)} + m_2^{(6)} \\ -m_3^{(1)} + m_3^{(2)} & -m_3^{(3)} + m_3^{(4)} & -m_3^{(5)} + m_3^{(6)} \end{pmatrix},$$

$$y = \frac{1}{6} \sum_{i=1}^{6} m^{(i)}.$$

Therefore, in this case, it is no longer required to invert a matrix for the calibration. Instead, only very few arithmetic operations on the measured data are needed, which is very computationally cheap and can be done without extra software on the user device itself.

3.4. Calibration procedure

We propose a simple method for obtaining six different measurements in six canonical orientations of the user device. This procedure depends on the assumption that the device is roughly box-shaped. If this is not the case, it has to be held in place with a suitable construction.

For this procedure, we simply have the user put the device in its six standard positions. In order to keep the device well-oriented and stable, simple everyday objects like window frames or bookshelves can be used. In a laboratory setting, nivellation of the involved surface can be performed beforehand to increase the accuracy.

In each of the six positions, the accelerometer output will be measured for a few seconds and then averaged in order to reduce the measurement noise. This finally gives us six vectors $m^{(1)}, \ldots, m^{(6)}$ with known true accelerations $a^{(1)}, \ldots, a^{(6)}$ resulting from Earth gravity.

An example of how the procedure can be performed is shown in the pictures below.

3.5. Examples

We were given two example datasets taken from two different devices and ran our method on this input data. One more dataset was captured by ourselves in order to test the calibration procedure. In the following, we describe the results of these experiments.

3.5.1. Example: Huawei

We have 29 accelerations $m^{(i)}$ which were measured in different positions. From these, the first six,

$$\begin{array}{lll} -9.54983 & 0.37829 & -0.999283 \\ 10.145 & 0.314528 & -1.37789 \\ 0.413309 & -9.47641 & -1.62076 \\ 0.368766 & 10.1442 & -0.910232 \\ 0.510292 & 0.338026 & -10.7851 \\ 0.231834 & 0.482186 & 8.60553 \\ \vdots & \vdots & \vdots \end{array}$$

are known to have been taken in the six standard positions described in Subsection 3.4. Our algorithm computes the model parameters

$$X = \begin{pmatrix} 1.00381 & -0.00227028 & -0.0141925 \\ -0.00324982 & 1.00003 & 0.00734762 \\ -0.019297 & 0.0362144 & 0.988311 \end{pmatrix},$$

$$y = \begin{pmatrix} 0.353222 \\ 0.363473 \\ -1.18129 \end{pmatrix}.$$

We point out that the computed offset errors y are within the tolerances from the sensor data sheet [5] that we were provided with, namely, ± 0.49 m/s^2 for the x and y axes and ± 1.96 m/s^2 for the z axis.

In order to determine how well the linear model fits the given data, we compute the relative fitting error for the first six measurements,

$$\max_{i=1,\ldots,6} \frac{\left\| F^{-1}(m^{(i)}) - a^{(i)} \right\|}{\left\| a^{(i)} \right\|} = 0.010601.$$

If one computes the same errors based on the raw measurements, one finds a much larger error,

$$\max_{i=1,\ldots,6} \frac{\left\| m^{(i)} - a^{(i)} \right\|}{\left\| a^{(i)} \right\|} = 0.17386.$$

The observed improvement after calibration is therefore about 16.4 times.

For the remaining measurements, we do not know the true accelerations. However, we know that they were taken in stationary positions and thus their norms should be g. After calibration, the maximum relative error in the norms is

$$\max_{i=1,\ldots,29} \frac{\left| \| F^{-1}(m^{(i)}) \| - \mathrm{g} \right|}{\mathrm{g}} = 0.0315425,$$

whereas for the original data we have

$$\max_{i=1,\dots,29} \frac{\left|\|m^{(i)}\| - \mathrm{g}\right|}{\mathrm{g}} = 0.122075.$$

Here the improvement is about 3.87 times.

3.5.2. Example: GT

For a second dataset labelled "GT", we were given 27 measurements where again the first six were used to calibrate the device. For this data, the algorithm computed the parameters

$$X = \begin{pmatrix} 0.990908 & 0.0193084 & -0.0191228 \\ -0.000163521 & 0.981047 & -0.00689508 \\ 0.0228047 & 0.00124993 & 0.998542 \end{pmatrix},$$

$$y = \begin{pmatrix} 0.482181 \\ 0.0587712 \\ 0.0440956 \end{pmatrix}.$$

The obtained offset errors are noticeably smaller than in the first example, indicating that this device has more accurate sensors or was better calibrated already in the factory.

As in the previous example, we compute the calibration error

$$\max_{i=1,\dots,6} \frac{\left\|F^{-1}(m^{(i)}) - a^{(i)}\right\|}{\left\|a^{(i)}\right\|} = 0.0158458$$

and compare it to the original sensor data,

$$\max_{i=1,\dots,6} \frac{\left\|m^{(i)} - a^{(i)}\right\|}{\left\|a^{(i)}\right\|} = 0.0725016.$$

The improvement is about 4.58 times.

We also again compute the relative errors in the norms,

$$\max_{i=1,\dots,27} \frac{\left|\|F^{-1}(m^{(i)})\| - \mathrm{g}\right|}{\mathrm{g}} = 0.0138985,$$

and compare it to the original data,

$$\max_{i=1,\dots,27} \frac{\left|\|m^{(i)}\| - \mathrm{g}\right|}{\mathrm{g}} = 0.0582853.$$

The improvement is about 4.19 times.

3.5.3. Test of robustness of calibration procedure

As suggested by our industry partner, we have compared the results of several calibration runs. For this, we took 20 measurements of five seconds each in each of the six standard positions with another Huawei phone, giving us 20 independent datasets with which we can perform calibration.

The mean of the model parameters over the 20 trials is

$$\begin{pmatrix} 1.00367 & -0.00200432 & 0.0211193 \\ 0.00396457 & 1.00055 & -0.00489847 \\ 0.00346578 & 0.00534244 & 0.989011 \end{pmatrix}, \quad \begin{pmatrix} 0.264937 \\ 0.418299 \\ -1.1462 \end{pmatrix}.$$

The standard deviations of the model parameters over these 20 trials are

$$\begin{pmatrix} 0.00021274 & 0.0016114 & 0.000203203 \\ 0.00109465 & 0.000171741 & 0.000251313 \\ 0.000307081 & 0.000715155 & 0.000315258 \end{pmatrix}, \quad \begin{pmatrix} 0.00439571 \\ 0.00300421 \\ 0.00463018 \end{pmatrix}.$$

These deviations are well below the magnitudes of the obtained model parameters, which indicates that repeated calibration runs should generally result in very similar calibration parameters. In other words, the calibration procedure seems to be quite robust.

3.6. Conclusions

We have presented a calibration method which

- is easy to implement,
- requires only simple measurements which the end user can perform in a few minutes using no extra tools,
- is computationally very cheap,
- significantly reduces the error on the tested data sets,
- allows the reconstruction of the true accelerations in a known coordinate system,
- results in a model which remains valid in the non-stationary case.

Acknowledgment

The authors would like to express their sincere gratitude to Assoc. Prof. Kiril Alexiev and to Angel Ivanov – a representative of MM Solutions AD for their time and valuable advices, that have helped the present work a lot.

References

[1] Chatfield, A.B.: Fundamentals of High Accuracy Inertial Navigation, volume 174 of Progress in astronautics and aeronautics. American Institute of Aeronautics and Astronautics, Reston, VA, USA, 1997.

[2] Forsberg, T., Grip, N., Sabourova, N.: Non-Iterative Calibration for Accelerometers with Three Non-Orthogonal Axes and Cross-Axis Interference. Research Report No. 8, Department of Engineering Sciences and Mathematics, Division of Mathematics, Lulea University of Technology, 2012.

[3] Grip, N., Sabourova, N.: Simple Non-Iterative Calibration for Triaxial Accelerometers. Research Report No. 7, Department of Engineering Sciences and Mathematics, Division of Mathematics, Lulea University of Technology, 2011.

[4] Woodman, O.: An introduction to inertial navigation. Technical reports published by the University of Cambridge. ISSN 1476-2986. (2007)

[5] MPU-6000 and MPU-6050 Product Specification Revision 1.0. InvenSense Inc. (2010)

Prediction of sanding in subsurface hydrocarbon reservoirs

Hilary Ockendon, Stanislav Stoykov, Tomasz Zorawik,
Stefka Dimova, Ivan Georgiev, Natalia Kolkovska,
Mikhail Todorov, Daniela Vasileva

1. Introduction

Sand production in oil and gas wells can occur if the fluid velocity exceeds a certain value. Due to drilling operations, the mechanical stresses can exceed the load bearing capacity of the rock. As the local stresses exceed certain level, a certain amount of rock is fractured into sand. Then, the sand is carried by the fluid through the wellbore depending on the flow rate. The amount of the solids can be less than a few grams per cubic meter of reservoir fluid or an essential amount. In the later case erosion of the rock and removing sufficient quantities of rock can occur. This can produce subsurface cavities which collapse and destroy the well.

When sanding is unavoidable it is necessary to estimate the characteristics of the process. Our aim was to generate a simple one-dimensional local model, which predicts the volume of sanding, the radius and the porosity of the yielded zone. Such model will help the company in the development of complex 3D models.

2. Mathematical model

The model we have generated is based on the continuum model presented in the paper [1], assuming radial propagation of the yielded zone around the wellbore (see Fig. 1). First we quote only the main results of the modelling in [1], concerning the quantities we are interested in.

By assuming small variations of the porosity, the relationship between the radius $R(t)$ of the yielded zone and the cumulative solids production $S(t)$ is derived:

$$S(t) = \frac{(1+\alpha)(\phi_y - \phi_n) r_w^{\frac{2\alpha}{1+\alpha}}}{2(1-\phi_y)} \left(R(t)^{\frac{2}{1+\alpha}} - r_w^{\frac{2}{1+\alpha}} \right), \tag{1}$$

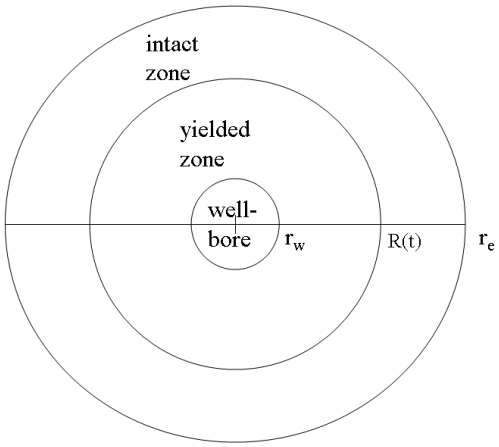


Figure 1: Wellbore, yielded and intact zones

where r_w is the radius of the wellbore, ϕ_n is the porosity of the intact zone, ϕ_y is the porosity of the yielded zone at $r = R(t)$, α is the coefficient of dilation.

From this equation we find:

$$\frac{dS(t)}{dt} = \frac{dR(t)}{dt}\,\frac{(\phi_y - \phi_n) r_w^{\frac{2\alpha}{1+\alpha}} R(t)^{\frac{1-\alpha}{1+\alpha}}}{(1-\phi_y)}. \tag{2}$$

Further, assuming continuity of the radial stress at the boundary between yielded and non-yielded zones, the following equation is obtained in [1]:

$$\frac{\beta\mu}{k}\,\frac{dS(t)}{dt}\left(1 - \frac{\ln\rho}{\ln\rho_e}\right) = \frac{p_c - p_w(t)}{\ln\rho} - 2\kappa + \beta\frac{p_e - p_w(t)}{\ln\rho_e} - \frac{2\alpha|\kappa'|}{r_w^2}\,\frac{(1-\rho^{-2})}{\ln\rho} S(t), \tag{3}$$

where $\rho = R(t)/r_w$, $\rho_e = r_e/r_w$, r_e is the radius of the reservoir (see Fig. 1), p_c is the critical wellbore fluid pressure, p_e is the reservoir pressure, β is the Biot's constant, μ is the viscosity of the fluid, k is the permeability, κ is the strength function.

The strength κ and the porosity are related by (see [1]):

$$\kappa(\phi) = \kappa(\phi_y) + |\kappa'|\,\frac{2\alpha(1-\phi_y)S(t)}{(1+\alpha) r_w^{\frac{2\alpha}{1+\alpha}} r^{\frac{2}{1+\alpha}}}. \tag{4}$$

For the evolution in time of the wellbore pressure $p_w(t)$ we have used

$$p_w(t) = \frac{p_e(1+\exp(-10t/T))}{2}, \quad T = 500 \times 24 \times 3600 \text{ sec.} \tag{5}$$

Equation (3) (together with the relations (1), (2), (4) and (5)), is an ordinary differential equation for the radius $R(t)$ of the yielded zone. The initial condition is

$$R(0) = r_w.$$

This initial value problem was solved in Maple and Mathematica by the fourth-fifth order Runge-Kutta method.

Once a solution for $R(t)$ is obtained in the time domain, the solids production $S(t)$ can be calculated from equation (1).

To find the porosity of the yielded zone, the second continuity equation from [1]

$$-\frac{\partial \phi}{\partial t} + \text{div}[(1-\phi)v_s] = 0 \tag{6}$$

has been simplified there by letting $\phi = \phi_y$. By using this simplification the following dependence of the porosity on the radial coordinate r and time t has been proposed in [1]:

$$\phi(r,t) = \phi_y - \frac{2\alpha(1-\phi_y)}{1+\alpha} \frac{S(t)}{r^{\frac{2}{1+\alpha}} r_w^{\frac{2\alpha}{1+\alpha}}}. \tag{7}$$

However, as we have found in the computational experiments, this formula gives non-physical behavior of the porosity, namely, it becomes negative for r close to r_w.

In the next section we derive another dependence for the porosity without making the above mentioned simplification. Besides we verify the expression (1) for $S(t)$.

3. Improvement of the expression for porosity

Let $\lambda = \dfrac{1-\alpha}{1+\alpha}$. From the equation for the solid velocity in [1]:

$$v_s(r,t) = -\frac{1}{r^{\frac{1-\alpha}{1+\alpha}}} \frac{q(t)}{r_w^{\frac{2\alpha}{1+\alpha}}}, \tag{8}$$

where $q(t)$ is the volumetric rate of solid production, we get

$$v_s = \frac{-q(t)}{r^\lambda r_w^{1-\lambda}}.$$

Now we rewrite the continuity equation (6) in polar coordinates and obtain

$$-\frac{\partial \phi}{\partial t} - \frac{q(t)}{r_w^{1-\lambda} r} \frac{\partial}{\partial r}\left(r^{1-\lambda}(1-\phi)\right) = 0.$$

After the substitution $u = r^{1-\lambda}(1-\phi)$ this equation takes the form

$$r^\lambda u_t - \frac{q(t)}{r_w^{1-\lambda}} u_r = 0.$$

This is a first order hyperbolic equation for u. Its characteristic curves are given by

$$\frac{dt}{r^\lambda} = -\frac{dr}{q(t) r_w^{\lambda-1}}.$$

Since $\int_0^t q(u)du = S(t)$ we get

$$S(t) + \frac{r^{\lambda+1}}{r_w^{\lambda-1}} \frac{1}{\lambda+1} = \text{const}.$$

The solution u is constant along the characteristics, thus:

$$S(t) + \frac{r^{\lambda+1}}{(\lambda+1) r_w^{\lambda-1}} = F\left((1-\phi) r^{1-\lambda}\right), \tag{9}$$

where F is an arbitrary sufficiently smooth function. Taking into account the condition

$$\phi = \phi_y \quad \text{on} \quad r = R(t)$$

we arrive at

$$S(t) + \frac{R(t)^{\lambda+1}}{(\lambda+1) r_w^{\lambda-1}} = F\left((1-\phi_y) R(t)^{1-\lambda}\right). \tag{10}$$

To relate $S(t)$ and $R(t)$ we use the equation for the mass balance at the boundary (from [1]):

$$(1-\phi_n)\frac{dR}{dt} = (1-\phi_y)\left(\frac{dR}{dt} - v_s\right)$$

and (8):

$$(\phi_y - \phi_n)R'(t) = \frac{(1-\phi_y)q(t)}{R^{\lambda} r_w^{1-\lambda}}.$$

Then we integrate with respect to t and use $R = r_w$ at $t = 0$ to get

$$S(t) = \frac{(\phi_y - \phi_n)(1+\alpha) r_w^{\frac{2\alpha}{1+\alpha}} \left(R^{\frac{2}{1+\alpha}} - r_w^{\frac{2}{1+\alpha}} \right)}{2(1-\phi_y)},$$

which verifies (1).

To find ϕ we substitute $S(t)$ in (10):

$$A\left(R^{\frac{2}{1+\alpha}} - r_w^{\frac{2}{1+\alpha}} \right) + BR^{\frac{2}{1+\alpha}} = F\left((1-\phi_y) R^{\frac{2\alpha}{1+\alpha}} \right),$$

where

$$A = \frac{(\phi_y - \phi_n)(1+\alpha) r_w^{\frac{2\alpha}{1+\alpha}}}{2(1-\phi_y)}, \quad B = \frac{(1+\alpha) r_w^{\frac{2\alpha}{1+\alpha}}}{2}.$$

We set

$$z = (1-\phi_y) R^{\frac{2\alpha}{1+\alpha}}$$

and obtain

$$F(z) = (A+B)\left(\frac{z}{1-\phi_y} \right)^{\frac{1}{\alpha}} - A r_w^{\frac{2}{1+\alpha}} \quad (\text{for } z > 0).$$

Now we go back to (9)

$$S(t) + B r^{\frac{2}{1+\alpha}} = (A+B) \frac{(1-\phi)^{\frac{1}{\alpha}} r^{\frac{2}{1+\alpha}}}{(1-\phi_y)^{\frac{1}{\alpha}}} - A r_w^{\frac{2}{1+\alpha}}.$$

Further we express $\phi(r,t)$ in terms of $R(t)$:

$$AR^{\frac{2}{1+\alpha}} + Br^{\frac{2}{1+\alpha}} = (A+B)\left(\frac{1-\phi}{1-\phi_y} \right)^{\frac{1}{\alpha}} r^{\frac{2}{1+\alpha}}.$$

Thus, solving this equation for $\phi(r,t)$, we get

$$\phi = 1 - \frac{1-\phi_y}{(A+B)^{\alpha}} \left[A\left(\frac{R(t)}{r} \right)^{\frac{2}{1+\alpha}} + B \right]^{\alpha}.$$

We substitute A and B and finally obtain the new formula for ϕ:

$$\phi(r,t) = 1 - \frac{1-\phi_y}{(1-\phi_n)^\alpha}\left[(\phi_y-\phi_n)\left(\frac{R(t)}{r}\right)^{\frac{2}{1+\alpha}} + (1-\phi_y)\right]^\alpha. \tag{11}$$

4. Numerical results

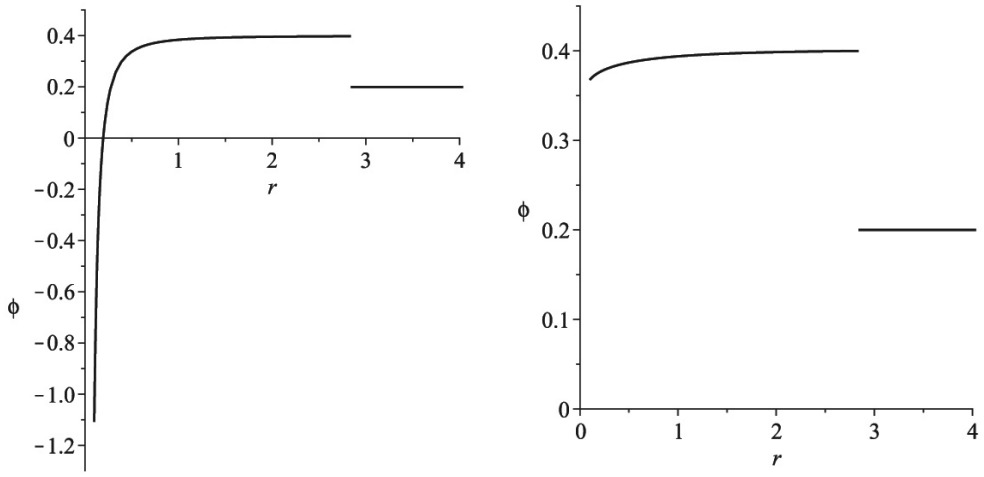
By using the described model we have investigated

- the evolution of the radius of the yielded zone $R(t)$,
- the cumulative solids production $S(t)$ and
- the porosity $\phi(r,t)$

for the following values of the model parameters:

$p_c = 4\times10^7$, $p_e = 4\times10^7$, $r_w = 0.1$, $r_e = 500$, $k = 10^{-14}$, $\mu = 10^{-3}$, $\phi_y = 0.4$, $\phi_n = 0.2$, $\beta = 0.9$, $\alpha = 0.01;\ 0.1$, $\kappa(\phi_y) = 10^5$, $|\kappa'| = 10^7;\ 10^8$.

First we show the porosity ϕ as a function of r at time $t = 500$ days, computed by formula (7) from paper [1] (Fig. 2, left), and by the corrected formula (11)



by formula (7) by the corrected formula (11)

Figure 2: Porosity $\phi(r,t)$ for $t = 500$ days and $\alpha = 0.01$, $|\kappa'| = 10^7$.

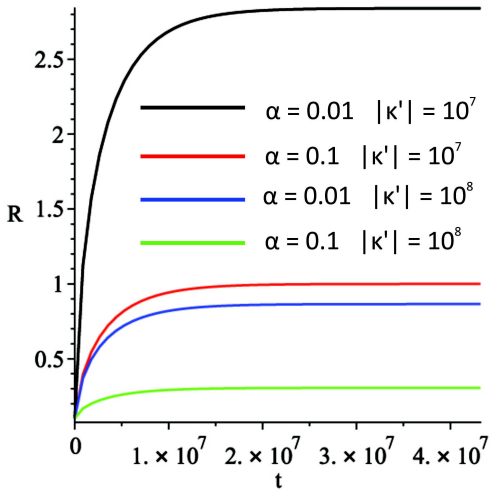


Figure 3: Evolution of the radius of the yielded zone for several values of α and $|\kappa'|$.

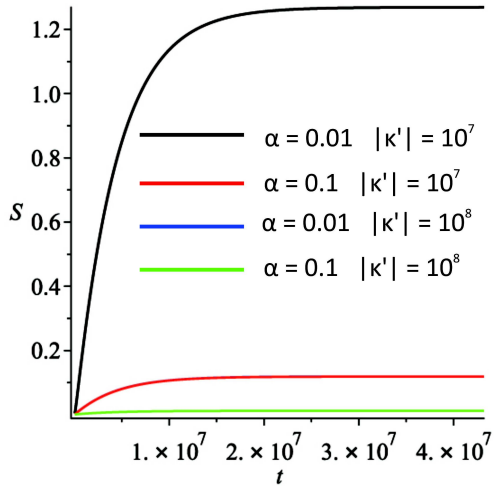


Figure 4: Cumulative solids production $S(t)$ for several values of α and $|\kappa'|$.

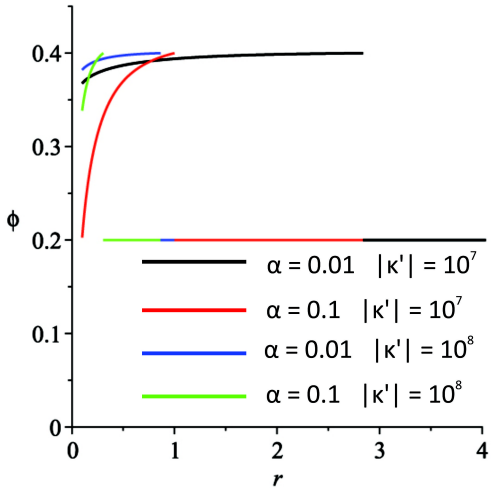


Figure 5: Porosity $\phi(r,t)$ for $t = 500$ days, for several values of α and $|\kappa'|$.

(Fig. 2, right). It can be seen that for small r, close to r_w, the porosity on Fig. 2, left, is negative.

The radius $R(t)$ of the yielded zone as a function of time is presented on Fig. 3, for four sets of parameters $\alpha > 0$ and $|\kappa'|$. It can be seen that the radius of the yielded zone increases at the initial time interval and then it reaches a steady-state value, thus the yielded zone stops growing. The steady-state value of the radius of the yielded zone depends on the model parameters.

The sand production $S(t)$ is obtained by using the computed radius $R(t)$ of the yielded zone and equation (1). It is presented on Fig. 4 for the same values of the parameters $\alpha > 0$ and $|\kappa'|$. The graphs of $S(t)$ for parameters $\alpha = 0.1,\ |\kappa'| = 10^7$ and $\alpha = 0.01,\ |\kappa'| = 10^8$ coincide within the plotting resolution.

Finally, the porosity is obtained from equation (11). It is shown on Fig. 5 for the same values of α and $|\kappa'|$. It can be seen that by using the new approximation (11) the porosity of the yielded zone is positive for $r \geq r_w$, moreover $\phi(r,t) \geq \phi_n$, which is physically realistic.

5. Conclusion

The model presented in [1] gives reasonable approximations for the radius $R(t)$ of the yielded zone and for the cumulative solids production $S(t)$, but it

does not give physically reasonable results for the porosity $\phi(r,t)$. With the new approximation of the porosity, the numerical results are physically reasonable.

The described model and its computer implementation can be used to predict the radius of the yielded zone $R(t)$, the cumulative solids production $S(t)$ and the porosity $\phi(r,t)$ for various scenarios.

The model can be improved taking into account the sand erosion due to abrasion. This could be done by including extra terms in the equations of continuity:

$$-\frac{\partial \phi}{\partial t} + \mathrm{div}[(1-\phi)v_s] = -c(v_f - v_s)\frac{\partial \phi}{\partial t},$$

$$\frac{\partial}{\partial t}(\phi\, c) + \mathrm{div}[\phi\, c\, v_f] = c(v_f - v_s)\frac{\partial \phi}{\partial t},$$

where c is the sand concentration in the fluid.

Alternative possibility is to construct elastoplastic model for the rock behavior.

References

[1] M. B. Gelikman, M. B. Dusseault, F. A. Dullien, Sand production as a viscoplastic granular flow, Society of Petroleum Engineers 27343 (1994), 41–50.