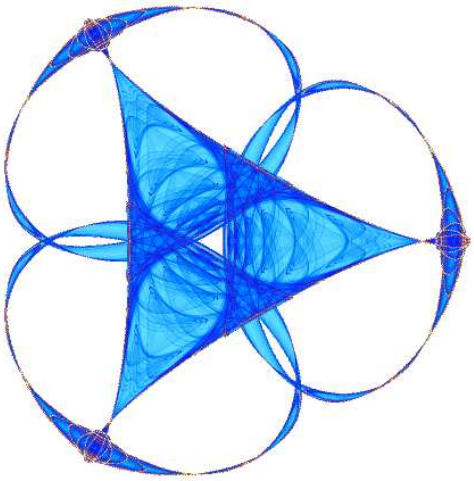
MODELING OF A NOVEL SOLUTION POTASH MINING PROCESS

By

Sergio Almada, Harvey Haugen
Hector Hernandez, Dori Luli
Sarath Sasi,
and
Lei Zhang

IMA Preprint Series # 2332

(September 2010)



INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA
400 Lind Hall
207 Church Street S.E.
Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370
URL: http://www.ima.umn.edu

Modeling of a Novel Solution Potash Mining Process

Sergio Almada[1], **Harvey Haugen**[2], **Hector Hernandez**[3], **Dori Luli**[4], **Sarath Sasi**[5], **Lei Zhang**[6]

1 Introduction

Potassium is one of the three basic plant nutrients along with nitrogen and phosphorus. Potash is extracted from buried ancient evaporites by underground or solution mining. This accounts for most of the potassium produced. In solution mining, water is used to dissolve a desired mineral from a geological ore zone into a solution through directional or vertical boreholes. Benefits of solution mining in comparison to conventional mining include lower capital expenditure requirements, operation scalability, quicker timeline to production, lower technical risk, and environmental impact.

We describe one of the current potash solution mining processes below. The process is initiated by drilling followed by a number of steps to develop a solution mining cavern. Water is injected to dissolve a sump area within the salt at the base of the solution mining cavern. The sump allows insoluble materials settle within the cavern to avoid affecting the solution mining process and the dissolved salt can be pumped out (Figure 1).

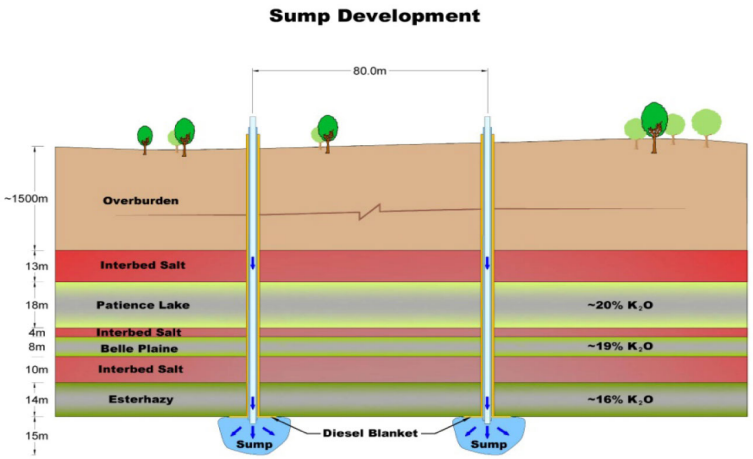


Figure 1: Sump Development

[1] Department of Mathematics, Georgia Institute of Technology
[2] Beechy Industries, Saskatchewan, Canada
[3] Software development, Center of Investigations in Mathematics (CIMAT)
[4] Department of Applied Math for the Life and Social Sciences, Arizona State University
[5] Department of Mathematics and Statistics, Mississippi State University
[6] Department of Mathematics, Purdue University

This is not a very efficient technique as most of the ore deposits lay in a horizontal layer. Beechy Industries proposed a novel solution mining process that is designed to increase the efficacy and reduce the cost. The process uses interconnected horizontal wells drilled through a high grade potash zone that should eventually dissolve out all the potash in the zone, while leaving the salt in the caverns created (see Figure 2). Potash will be dissolved out of the ore using a brine saturated in sodium chloride, but substantially undersaturated in potassium chloride. The mechanics of the whole process is similar to that of a meandering stream except for the free surface and the changing boundaries. Flow momentum will cause the degradation of walls near concave banks and the convex banks will advance because of sediment deposition. Once the process has been understood the initial configuration of the well can be optimized for the best output. A realistic model should incorporate the complex mechanism of fluid dynamics, sediment transport, and dissolution of the boundary. The final objective of this project is to use the results of this study to analyze the feasibility of the new technique.

Figure 2: Horizontal development

We plan to do the study in four stages.

(I) Develop an empirical model that uses the given data to simulate the horizontal development.

(II) Develop a mathematical model.

(III) Numerical simulation.

(IV) Propose future work.

In section 2 we describe some of the results from the empirical data. We give the sketch of what could be a comprehensive model in section 3. Then we go on to describe the depth averaged 2-D model that we implemented. In section 4 we describe the numerical techniques and results from the simulations.

2 Empirical Results

In this section we use some of the data provided by Beechy Industries to test if the assumptions about the development of the well will hold. In this direction we ran a simulation of the steady state Navier-Stokes in a curved geometry. And as expected the velocity profile shows a higher velocity on the outer banks and lower velocity on the inner banks (see Figure 3). In the next set of figures we use the dissippation rate in the given data to plot the horizontal development of the wells.

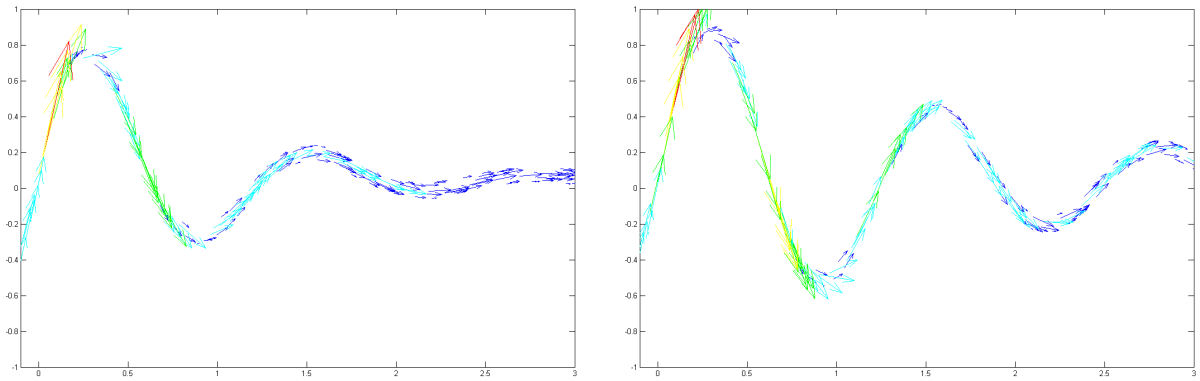


Figure 3: Velocity profile in the channel

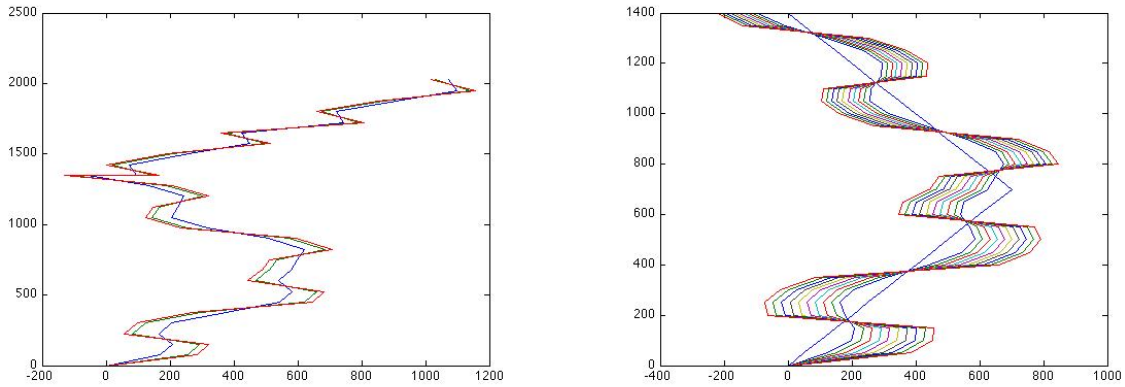


Figure 4: Channel Widening

3 The Mathematical Model

The mathematical model should be able to describe the fluid flow, the heat transport, the dissolution, the formation and movement of the salt sediments, and the change in the geometry. The model should be able to keep track of the velocity field, the density, the temperature, the concentration of potash and salt, and the sediment dynamics. The most important aspects of the problem are the expansion due to dissolution and the contraction due to sedimentation. Before formulating the model we look at the space and time scales involved. The ratio between the length of the horizontal wells and its radius is very large, therefore it is computationally very expensive to capture the effects of the curvature on the well development under the cartesian coordinate system. Hence, it is reasonable that the model formulated be transformed to a curvilinear coordinate system. Also, it is reasonable to assume that the changes in geometry happen at a larger time scale than the other phenomena of interest. Therefore, we will not be coupling the change in geometry with the other equations.

The fluid flow can be described by the Navier-Stokes equations for mass and momentum

conservation with the appropriate boundary conditions

$$\begin{aligned}\frac{\partial}{\partial t}(\rho) + \nabla.(\vec{u}\rho) &= S_1(C_p, C_s, T, \vec{u}) \\ \frac{\partial}{\partial t}(\rho\vec{u}) + \vec{u}.\nabla(\rho\vec{u}) &= -\nabla p + \mu\nabla.(\nabla(\rho\mu)) + \vec{F}.\end{aligned}$$

Here $S_1(C_p, C_s, T, \vec{u})$ is the source term that accounts for the dissolving salt and potash. We apply the no-slip boundary condition on the walls of the well, i.e. $u|_{wall} = 0$. The influx is a constant.

Now we need to model the dynamics of the potash and salt in the system

$$\begin{aligned}\frac{\partial}{\partial t}(C_p) + (\vec{u}.\nabla)C_p &= S_2(C_p, C_s, T, \vec{u}) + D_p\Delta C_p \\ \frac{\partial}{\partial t}(C_s) + (\vec{u}.\nabla)C_p &= S_3(C_p, C_s, T, \vec{u}) + D_s\Delta C_s.\end{aligned}$$

Here again $S_2(C_p, C_s, T, \vec{u})$ and $S_3(C_p, C_s, T, \vec{u})$ are the source terms for the potash and salt, respectively. It is to be noted that all the source terms are related. We also have $\frac{\partial}{\partial_n}(C_p)|_{wall} = S_2(C_p, C_s, T, \vec{u})$ and $\frac{\partial}{\partial_n}(C_s)|_{wall} = S_3(C_p, C_s, T, \vec{u})$.

The heat transfer part of the model is given by $\frac{\partial T}{\partial t} = D_H\nabla^2 T - \vec{u}\cdot\nabla T$ with the boundary condition given by $k_T\frac{\partial}{\partial_n}T|_w all = h_T(T - T_r)$ where k_T is the conductivity and h_T is the heat transfer coefficient.

Since there are practically no suspended sediments we consider only the bed transport which can be modeled in a heuristic manner. The sediment size is in the range of $2 - 5$ mm. We could select the Meyer-Peter and Muller transport formula that is valid for uniform sediment with a mean particle size ranging from 0.23 to 28.6 mm. The transport formula is $q_s* = 8\left(\tau_* - \tau_{*c}\right)^{3/2}$. The formula uses the following nondimensionalization for shear stress τ and for sediment volumetric discharge per unit width q_s

$$\begin{aligned}\tau_* &= \frac{\tau}{(\rho_s - \rho)(g)(d)} \\ q_{s*} &= \frac{q_s}{d\sqrt{\frac{\rho_s - \rho}{\rho}gd}} = \frac{q_s}{Re_p\nu}.\end{aligned}$$

Here d is the average grain diameter, ρ_s and ρ are the densities of the fluid and sediment respectively. For fluids with large Reynold's number, we have $\tau_{*c} = 0.047$.

As for the geometry, a heuristic model has to be developed based on the empirical data. Transforming to the curvilinear coordinate system $(x, y, z) \mapsto (\xi_1, \xi_2, \xi_3)$, we expect that $\frac{\partial}{\partial t}R(\xi_1, \xi_2, \xi_3) \sim S_1$. Establishing the right relation will be one of the challenges in simulating the process.

Due to various constraints, such a comprehensive model is not within the scope of this project, we consider a much simplified model after making the following assumptions:

1. The well is a straight channel.

2. The system is isothermic.

3. The system has only potash in it.

4. Since the Reynold's and Peclet numbers are high, the diffusion terms can be ignored.

5. The pressure gradient nullifies the effect of other forces.

Under these assumptions the density of the fluid ρ depends only on the concentration of potash. Assuming the dependence to be smooth, we can use the first order approximation

$$\rho(C) = \rho_0 + \frac{\partial}{\partial t}\rho(0)C + O(C^2).$$

The data we have suggest that it is appropriate to make the so called Boussinesq approximation by dropping the quadratic term to give

$$\rho = \rho_0 + \frac{\partial}{\partial t}\rho(0)C, \quad C \to 0. \tag{1}$$

The new governing equations are

$$\begin{aligned}
\frac{\partial}{\partial t}(\rho) + \frac{\partial}{\partial x}(\rho u_x) + \frac{\partial}{\partial y}(\rho u_y) + \frac{\partial}{\partial z}(\rho u_z) &= S_1(C_p, \vec{u}) \\
\frac{\partial}{\partial t}(C_p) + \frac{\partial}{\partial x}(u_x C_p) + \frac{\partial}{\partial y}(u_y C_p) + \frac{\partial}{\partial z}(u_z C_p) &= S_2(C_p, \vec{u}) \\
\frac{\partial}{\partial t}(\rho u_x) + \frac{\partial}{\partial x}(\rho u_x^2) + \frac{\partial}{\partial y}(\rho u_x u_y) + \frac{\partial}{\partial x}(\rho u_x u_z) &= 0 \\
\frac{\partial}{\partial t}(\rho u_y) + \frac{\partial}{\partial x}(\rho u_y^2) + \frac{\partial}{\partial y}(\rho u_x u_y) + \frac{\partial}{\partial x}(\rho u_y u_z) &= 0 \\
\frac{\partial}{\partial t}(\rho u_z) + \frac{\partial}{\partial x}(\rho u_z^2) + \frac{\partial}{\partial y}(\rho u_y u_z) + \frac{\partial}{\partial x}(\rho u_x u_z) &= 0.
\end{aligned}$$

3.1 The Depth-Averaged 2-D Model

The depth-averaged 2-D model can simulate the effects of large-scale roughness structures, such as channel contraction, expansion, and curvature on the flow field, using fine meshes. In addition, the depth-averaged 2-D model can account for the effect of channel banks through boundary conditions and considers the effect of horizontal turbulent diffusion through the eddy viscosity. The depth-averaged quantity ϕ of a three-dimensional variable ϕ is defined by

$$\phi = \frac{1}{h}\int_{z_b}^{z_s} \phi \, dz, \tag{2}$$

where z_b stands for the flow bed and z_s for the flow surface. Integrating the continuity equation

$$\frac{\partial}{\partial t}(\rho) + \frac{\partial}{\partial x}(\rho u_x) + \frac{\partial}{\partial y}(\rho u_y) + \frac{\partial}{\partial z}(\rho u_z) = S_1(C_p, \vec{u})$$

over the flow depth yields

$$\int_{z_b}^{z_s} \frac{\partial}{\partial t}\rho dz + \int_{z_b}^{z_s} \frac{\partial}{\partial x}(\rho u_x)dz + \int_{z_b}^{z_s} \frac{\partial}{\partial y}(\rho u_y)dz + \int_{z_b}^{z_s} \frac{\partial}{\partial z}(\rho u_z)dz = \int_{z_b}^{z_s} S_1(C_p, \vec{u})dz,$$

which is reformulated using the Leibniz integral rule as

$$\frac{\partial}{\partial t}(\rho\int_{z_b}^{z_s}dz)+\frac{\partial}{\partial x}(\rho\int_{z_b}^{z_s}u_x dz)-\rho u_{hx}\frac{\partial z_s}{\partial x}+\rho u_{bx}\frac{\partial z_b}{\partial x}+\frac{\partial}{\partial y}(\rho\int_{z_b}^{z_s}u_y dz)-\rho u_{hy}\frac{\partial z_s}{\partial y}+\rho u_{by}\frac{\partial z_b}{\partial y}+\rho u_{hz}-\rho u_{bz}=\tilde{S}_1(C_p,\vec{u}). \tag{3}$$

Substituting non slip boundary conditions

$$u_{bx}=0, u_{by}=0, u_{bz}=0$$

and the kinematic condition

$$\frac{\partial z_s}{\partial t}+u_{hx}\frac{\partial z_s}{\partial x}+u_{hy}\frac{\partial z_s}{\partial y}=u_{hz}$$

into (**??**) leads to the depth-integrated 2-D continuity equation

$$2\frac{\partial}{\partial t}(\rho h)+\frac{\partial}{\partial x}(\rho h U_x)+\frac{\partial}{\partial y}(\rho h U_y)=\tilde{S}_1(C_p,\vec{u}),$$

where U_x and U_y are the depth-averaged quantities of local velocities u_x and u_y defined by (**??**). Repeating this procedure for the other equations we get

$$\begin{aligned}
2\frac{\partial}{\partial t}(\rho h)+\frac{\partial}{\partial x}(\rho h U_x)+\frac{\partial}{\partial y}(\rho h U_y) &= \tilde{S}_1(C_p,\vec{u})\\
\frac{\partial}{\partial t}(hC_p)+\frac{\partial}{\partial x}(hU_xC_p)+\frac{\partial}{\partial x}(hU_yC_p) &= \tilde{S}_2(C_p,\vec{u})\\
\frac{\partial}{\partial t}(\rho h U_x)+\frac{\partial}{\partial x}(\rho h U_x^2)+\frac{\partial}{\partial y}(\rho h U_x U_y)+\frac{\partial}{\partial x}(\rho D_{uu})+\frac{\partial}{\partial y}(\rho D_{uv}) &= 0\\
\frac{\partial}{\partial t}(\rho h U_y)+\frac{\partial}{\partial x}(\rho h U_x U_y)+\frac{\partial}{\partial y}(\rho h U_y^2)+\frac{\partial}{\partial x}(\rho D_{uv})+\frac{\partial}{\partial y}(\rho D_{vv}) &= 0,
\end{aligned}$$

where $D_{uu}=\int_{z_b}^{z_s}(u_x-U_x)^2dz$, $D_{uv}=\int_{z_b}^{z_s}(u_x-U_x)(u_y-U_y)dz$, and $D_{vv}=\int_{z_b}^{z_s}(u_y-U_y)^2dz$. The final step is to model the source terms $\tilde{S}_1(C_p,\vec{u})$ and $\tilde{S}_2(C_p,\vec{u})$. Since the diffusion term from the governing equation for C_p has been taken off, we define $\tilde{S}_1(C_p,\vec{u})=\tilde{S}_2(C_p,\vec{u}):=\rho q_d=k(C_\infty-C)$ where $k=\frac{D}{h}\Big(\frac{uh}{\nu}\Big)^{0.8}\Big(\frac{\nu}{D}\Big)^{\frac{1}{3}}$. Here D is the diffusion coefficient of potash and ν is the kinematic viscosity. In a straight channel directed along the x-axis, $U_y\approx 0$. Hence, we are left with the following set of equations

$$\begin{aligned}
2\frac{\partial}{\partial t}(\rho h)+\frac{\partial}{\partial x}(\rho h U_x) &= \rho q_d\\
\frac{\partial}{\partial t}(hC_p)+\frac{\partial}{\partial x}(hU_xC_p) &= \rho q_d\\
\frac{\partial}{\partial t}(\rho h U_x)+\frac{\partial}{\partial x}(\rho h U_x^2) &= 0.
\end{aligned}$$

4 Numerical Results

The model given in the previous section has been implemented using two different techniques, the finite difference method and the spectral method, so that the results can be compared.

4.1 Finite difference method

In order to approximate our set of partial differential equations, we use the finite difference method. Then we approximate the partial derivatives with respect to time by

$$\frac{\partial h}{\partial t} \approx \frac{h_x^{t+\Delta t} - h_x^t}{\Delta t},$$

while the partial derivatives with respect to the space dimension are approximated as

$$\frac{\partial h}{\partial x} \approx \frac{h_{x+\Delta x}^t - h_x^t}{\Delta x},$$

or by

$$\frac{\partial h}{\partial x} \approx \frac{h_x^t - h_{x-\Delta x}^t}{\Delta x},$$

depending on the flow direction, commonly known as the *up-winding* effect. Here Δt and Δx are the time and space steps of the finite difference discretization. We apply the first order finite difference method to simulate the height, the velocity, and the concentration in our model. We use $h(x, t = 0) = -x + 1$, $u(x, t = 0) = -x + 1$, and $C(x, t = 0) = 0$ as initial conditions for $x = [0, 1]$. We apply boundary conditions only at $x = 0$ for $h(x = 0, t > 0) = 1$, $u(x = 0, t > 0) = 1$, and $C(x = 0, t > 0) = 0$. We add a source term, $S_C = k\,(C_\infty - C)$, to the concentration in the whole domain. Due to the presence of shocks, the computed solution becomes out of bound, yielding unexpected results.

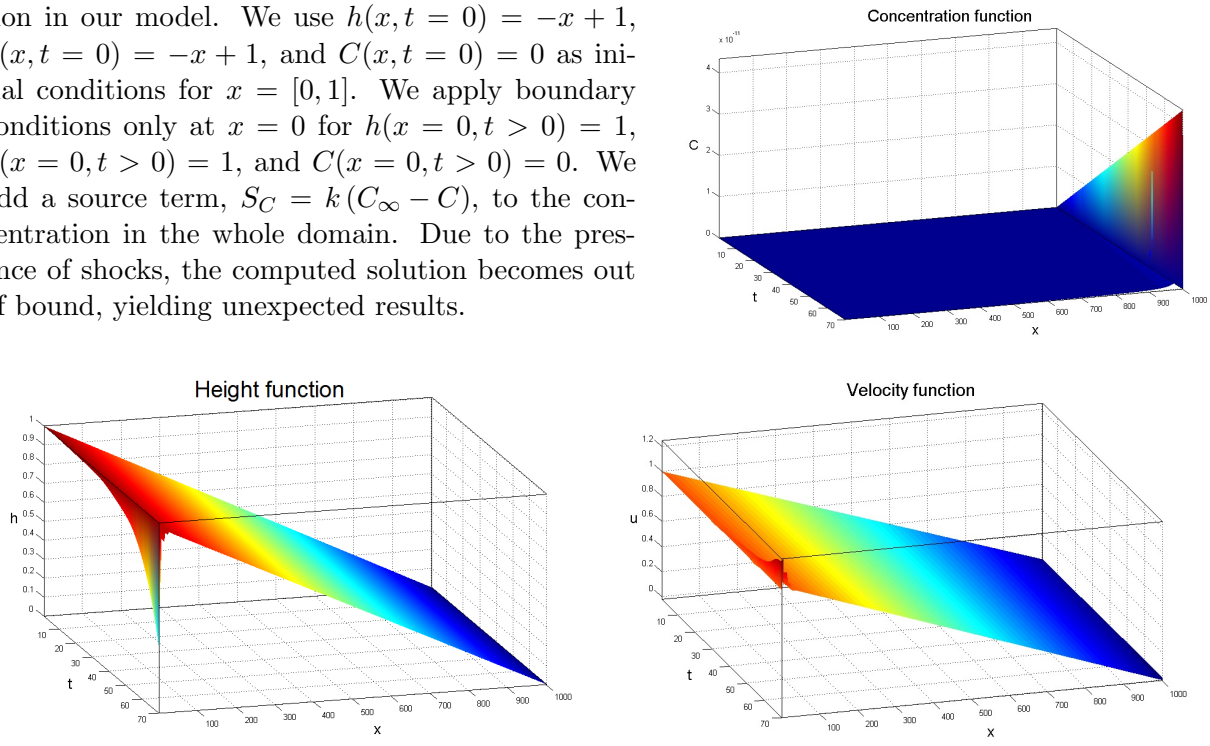


Figure 5:

4.2 Spectral method

Shallow Water Assumption

We will only consider the two dimensional equation. The model we focus on solving will include velocity field, concentration, and height. The main objective will be to study how the channel changes. In order to make several simplifications, we simulate the 2-dimensional velocity field on a boundary region similar to the ones the original problem is suppose to tackle.

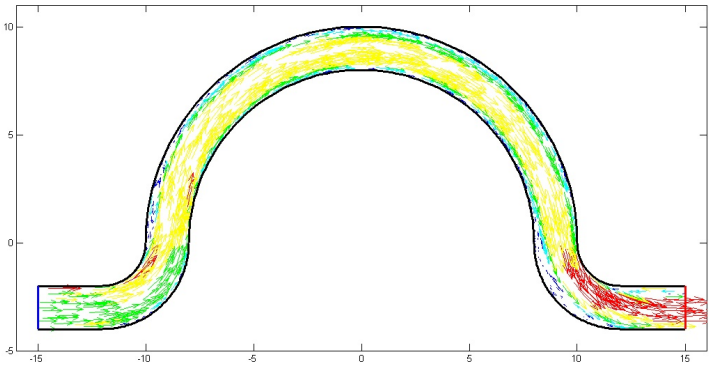


Figure 6: Simulation of the velocity field without potash in an approximate pipe

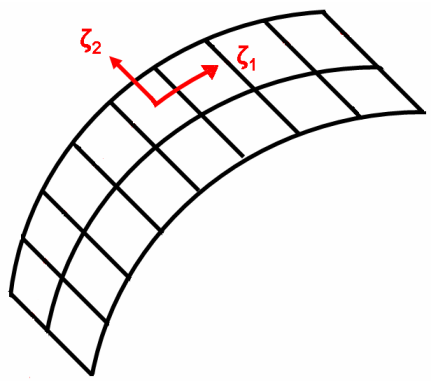


Figure 7: Parallel flow

Figure **??** shows that the flow is locally perpendicular to the tangent at each point (see figure **??**). Since our first simplification was to consider a straight channel, we drop the velocity component in y. This simplification is common in the literature and is called the shallow water assumption.

Large Reynolds number

We further notice that the Reynolds number for our system is large. So, as a further simplifi-

cation we drop the diffusive terms in each equation. Hence, we only consider

$$\begin{aligned}\frac{\partial}{\partial t}(\rho h)+\frac{\partial}{\partial x}(\rho h U_x) &= q_d\\ \frac{\partial}{\partial t}(hC_p)+\frac{\partial}{\partial x}(hU_xC_p) &= \rho q_d\\ \frac{\partial}{\partial t}(\rho h U_x)+\frac{\partial}{\partial x}(\rho h U_x^2) &= 0.\end{aligned}$$

In order to establish boundary conditions, we assume constant flow getting into the horizontal region. Hence, we establish the boundary conditions

$$\begin{aligned}u(t,0)h(t,0) &= u(0,0)h(0,0), \quad t>0\\ c(t,0) &= c(0,0).\end{aligned}$$

This is the problem we will tackle numerically.

4.2.1 Numerical Background

The idea of this implementation is to find an efficient way to compute $1-$dimensional spatial derivatives and then do some discretization on the time derivative. Since the time derivative is also coupled, this is not as straight forward and requires some calculations.

Using the product formula to open the crossed terms in the time derivative for the first two equations, we get the system

$$\begin{aligned}\rho\partial_t h + h\partial_t\rho &= q_d - \partial_x(\rho h u)\\ C\partial_t h + h\partial_t C &= \rho q_d - \partial_x(Chu),\end{aligned}$$

from which we can solve for both $\partial_t C$ and $\partial_t h$. The result is

$$\partial_t h = \frac{\rho_0}{(1-\rho_1\rho)}q_d - \frac{1}{\rho_0}\partial_x(\rho h u) + \frac{\rho_1}{\rho_0}\partial_x(Chu)$$

and

$$\partial_t C = \left(\rho q_d - \partial_x(Chu) - C\partial_t h\right)/h.$$

Likewise, from the third equation, we can write $\partial_t u$ in terms of the other time and spatial derivatives:

$$\partial_t u = -\frac{1}{\rho h}\left(\partial_x(\rho h u^2) + \rho u\partial_t h + \rho_1 u h\partial_t C\right).$$

We partition the time interval uniformly $t_j = jT/N$, where N is the number of time steps (hence $dt = 1/N$ is the step in time). So, if we were to implement finite differences and we, for a general function of time f, denote $f^j = f(t_j)$, we would have the first approximation in time

$$h^j = h^{j-1} + 2dt\left\{\frac{1}{\rho_0}(1-\rho_1\rho^{j-1})q_d^{j-1} - \frac{1}{\rho_0}\partial_x(\rho^{j-1}h^{j-1}u^{j-1}) + \frac{\rho_1}{\rho_0}\partial_x(C^{j-1}h^{j-1}u^{j-1})\right\} \tag{4}$$

$$C^j = C^{j-1}\left(\rho^{j-1}q_d - \partial_x(C^{j-1}h^{j-1}u^{j-1}) - C^{j-1}\partial_t h\right)/h^{j-1} \tag{5}$$

$$u^j = u^{j-1} - \frac{1}{\rho h}\left(\partial_x(\rho^{j-1}h^{j-1}(u^{j-1})^2) + \rho u^{j-1}\partial_t h + \rho_1 u^{j-1}h^{j-1}\partial_t C\right), \tag{6}$$

where all the derivatives on the right hand side are to be replaced by $\partial_t f = (f^j - f^{j-1})/(2dt)$ in the last expression. So, we are left to first design a procedure to derivate, and to improve this method of stepping in time.

Let us start with the stepping in time by the so called Runge-Kutta methods.

4.2.2 Time stepping

We explain the algorithm in an abstract sense, so that we don't have to carry all the messy expressions of the model. Suppose a function f is smooth and depends on a single 1-dimensional parameter. Then, a good approximation up to order k is given by The taylor expansion

$$f^j + \partial_t f^j dt + \frac{\partial_t^2 f^j}{2!}(dt)^2 + ... + \frac{\partial_t^k f^j}{k!}(dt)^k.$$

The idea of this method is to program an approximation for such an expansion. This expression can be rewritten recursively. For illustrative purposes we do the later, up to order 2

$$f^j + \partial_t \left(f^j + \frac{\partial_t}{2}(f^j + \frac{\partial_t}{3} f^j) \right).$$

With this expression we can adjust the order of the time approximation with an almost effortless expression.

4.2.3 Space Derivatives

The computation of the spatial derivative is based on a spectral method transformation. To explain this, suppose we can write our function f as

$$f(t,x) = \sum_1^N f_n(t)\varphi_n(x),$$

where $\{\varphi_n\}_n$ is an orthonormal frame (for example Chebychev polynomials). Then, the derivative is simply given by

$$\partial_x f(t,x) = \sum_1^N f_n(t)\partial_x\varphi_n(x).$$

For most families of frames $\mathcal{F}$, the terms $\partial_x\varphi_n(x)$ can be rewritten as a finite combination of elements in $\mathcal{F}$. Hence, allowing us to write a simple closed formula for the coefficients of the derivative.

4.2.4 Fourier Techniques

Our first approach was to use a Fourier basis. In this case there is a huge collection of algorithms available (the so called fast Fourier transform being the most celebrated one). At the same time, it has several disadvantages in terms of implementation. The first, and probably most important one, has to do with aliasing and aspects related with Gibbs phenomena.

Let f be the Fourier transform F. Then, the derivative is given by $i\omega F(\omega)$. When programming this formula, we need to be take several steps in order to smooth out all possible spectral spikes. The procedure is:

1. Discrete Fourier transform assumes that the function to be transformed is periodic. This can create sharp discontinuities. So, the first step is to add some additional zeros to the samples of the function.

2. The discrete Fourier transform gives the positive frequencies on the first half of the data, but the other half is concatenated to it. We need to take it into account and consider the non-symmetric identity function in this sense.

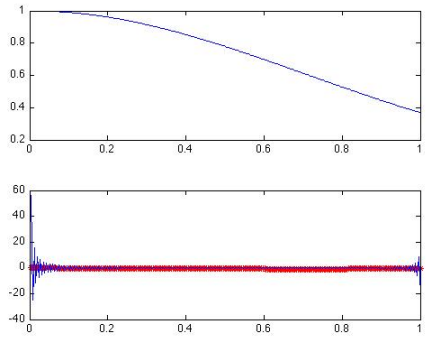


Figure 8: A function together with the exact derivative and the oscillatory Fourier derivative

3. Since the inverse Fourier transform is going to assume the resulting frequency domain function is periodic, we need to add a window. We choose the super exponential window e^{w^2/σ^2} because it is invariant under Fourier transformation, and because, theoretically speaking, it corresponds to the standard mollifier.

Although this method proves itself to be accurate to compute derivatives, we decide not to use it in the final numerical implementation. This is so, since we expect to have waves moving throughout the (non-linear) characteristic. In figure **??**, we can see the exact derivative and the derivative computing using the Fourier method. We see how strong (with respect to the magnitude of the derivative) oscillations appear. This is the case because when making the original function periodic, some sharp edges appear.

4.2.5 Chebychev's Method

After trying the Fourier method, we follow a similar idea, but instead of using trigonometric functions, we use polynomials. In particular, we take Chebychev polynomials and use polynomial interpolation. In this manner, we drop the periodicity requirement. But we cannot take uniform grids in space anymore.

While doing any kind of interpolation, it is standard to simply consider uniform points. In this case it turns out that this scheme breaks. Indeed, is not only that the interpolation does not converge as the number of points, N, increases, but the error can be as big as 2^N. There are several sets of points that are used, they all share in common the fact that as $N \to \infty$, they are distributed with density of the order

$$\frac{N}{\pi\sqrt{1-x^2}}.$$

In particular the average spacing is $O(N^{-2})$ for x near 1 and $O(N^{-1}$ in the interior of $(0,1)$.

For our computations, we chose the so called Chebychev points

$$x_j = cos(j\pi/N), \quad j = 0, ..., N.$$

Geometrically, Chebychev points are the projections on $[-1,1]$ of equidistant points in the upper half of the unit circle.

The output of this procedure is actually a matrix that codes the linear operation of differentiation. After some calculations, the shape of this derivative is given. We encourage the reader to see

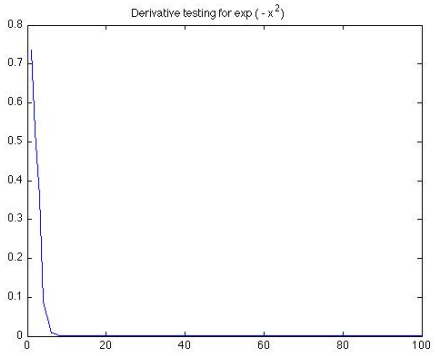


Figure 9: Error between the exact derivative of e^{-x^2} and the one computed by Chebychev method

this shape on the book by Trefethen [**?**]. The code used is included in the appendix. We present the results in figures 9 and 10. They are the difference between the exact derivative computed analytically and the one computed with our code for the functions $f(x) = e^{-x^2}$ and $f = 1$. We are particularly interested in Figure 10 which shows that as the dimension of the matrix increases, the error tends to oscillate more. The reason for that is due to the polynomial approximation: if we have n points, we approximate with a degree n polynomial. From our testing and programming of

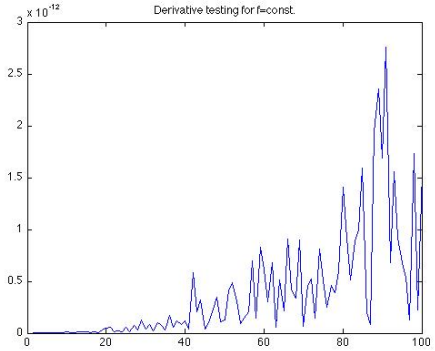


Figure 10: Error between the exact derivative of the constant function and the one computed by our Chebychev method

this code we learn some heuristics about this procedure:

- The initial data is interpreted to be sampled (according to the Chebychev scheme) from a polynomial interpolation.
- The result still has to be interpolated back, since it only gives the derivatives at the sampling points.

The second point turns out to be of importance in our particular case due to shock appearances. We simulate with initial velocity equals to 1 in each point x. The height is also constant and is equal to 1. Using this method, we run the same simulation with very low potash concentration at the beginning. The results (reported in Figure 11 and Figure 12) are consistent with intuition.

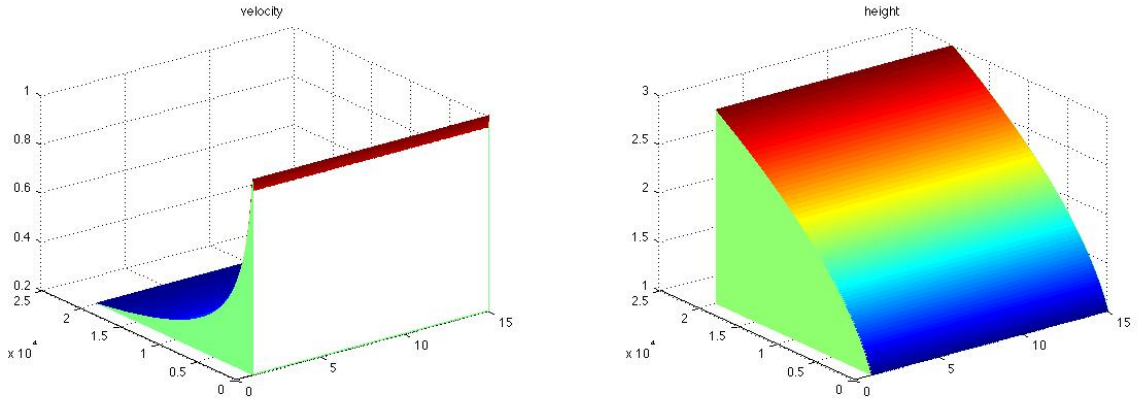


Figure 11: Velocity and Height when the water is with almost no potash

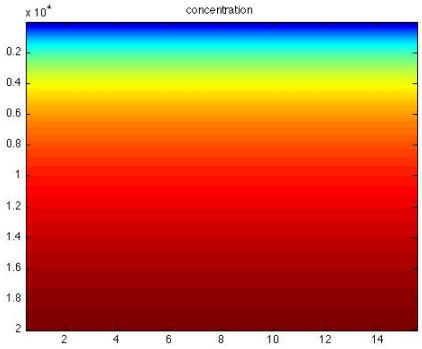


Figure 12: Concentration when we pump in water with almost no potash

- The height is increasing, as we can see from Figure 11. This is intuitive, since the flow is removing potash, which implies falling salt and hence channel widing.
- Since the channel width is increasing and the boundary conditions states that the initial flow is constant, the velocity should decrease. We confirm this idea in Figure 11.
- Finally concentration is increasing in time since we are taking the potash from the environment. This is seen in the height diagram in Figure 12.

An important feature we have observed, is that we didn't run the simulation for long enough time to see a gradient in time. We let the simulation run, and the first picture we have gives us some evidence of gradient present in the concentration, which is presented in Figure 13. We might remark that after the appearance of this gradient a shock comes and our simulation is yet not sophisticated enough to track it.

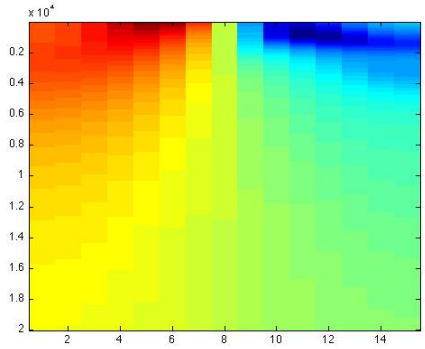


Figure 13: Spatial gradient in concentration

5 Future work

In this project we have only been able to take a preliminary look at the problem. In some discussions doubt was raised about the effectiveness of the depth averaging technique because of the cylindrical geometry of the mining wells. A more ambitious and precise work could begin from the sketchy 3D model introduced in Section 3. As the curvature of the well is one of the most significant aspects of the process, ideally one should start with a model in curvilinear coordinates. Another important aspect is the derivation of the source terms associated with dissipation. A better model should use the empirical data to construct a source term that improves on the current model.

6 Appendix

The code to implement the Fourier derivative is given by

```
function v_prime=Derivative_Fourier_Basic(v)

N=length(v); %% assumed of the form 2^n
N1=floor((N-1)/2);
N2=-(N/2)*ones(rem(N+1,2));
omega=sqrt(-1)*[ (0:N1)*exp( -(0:N1).^2 ) N2 (-N1:-1)*exp( -(-N1:-1).^2 )];

vhat=fft(v);
omega_vhat=omega.*vhat;
v_prime=real( ifft(omega_vhat) );
```

The code for the Chebychev derivative is given below.

```
function [D,grid]=DerivativeSpectralCheb(N)
x=cos(pi*(0:N)/N)';
X=repmat(x,1,N+1);
c=[2; ones(N-1,1);2].*(-1).^(0:N)';
dX=X-X';
D=( c*(1./c)' ) ./ ( dX + (eye(N+1)));
```

```
D=D-diag( sum(D') );
grid=x;
end
```

When implementing the code, we use a general order time stepping and a Chebychev matrix. After several scaling versions, the final code is:

```
function [u,h,c]=solvePDEBdry(uinitial, hinitial, cinitial, dt, T, approx)


L=length(uinitial);
x=cos( pi*(0:L-1)/L );

t=0:dt:T;
u(1,:)=uinitial;
h(1,:)=hinitial;
c(1,:)=cinitial;

bdry=u(1,1)*h(1,1);
bdryC=c(1,1);

for m=2:length(t)
haux=h(m-1,:);
uaux=u(m-1,:);
caux=c(m-1,:);

[partialh,partialu,partialc]=ComputePartial( u(m-1,:),h(m-1,:),c(m-1,:) );
for r=1:approx
haux=haux+(dt/(approx-(r-1) ) )* partialh;
uaux=uaux+(dt/(approx-(r-1) ) )* partialu;
caux=caux+(dt/(approx-(r-1) ) )* partialc;
end

h(m,:)=haux;
u(m,:)=uaux;
c(m,:)=caux;

    c(m,1)=bdryC;
    u(m,1)=bdry / h(m,1);
end



function [dh,du,dC]=ComputePartial( vel, he, con)
rho1=1.984e6;
rho0=.996 ;
```

```
rh=rho0+rho1*con;
q=qd( vel, he, con,rh);

dev_rhu= CalDerivative( rh.*vel.*he );
dev_Chu=CalDerivative( he.*con.*vel );
dev_rhu2=CalDerivative( rh.*he.*vel.*vel );

dh=( (1/rho0) * (1-rho1*rh) ).* q + (rho1/rho0)*dev_Chu - dev_rhu / rho0 ;
dC= (rh.* q)./he - dev_Chu./he-(con.*dh)./he;
du= -dev_rhu2./ ( rh.*he ) - rho1*( vel.* dC )./ rh - (vel.*dh)./he;

function q=qd(U,H,C,R)

D=1.9e-9;
nu=1e-6;
Kinfty= (D^.75) * nu^(1/3-.8);
Cinfty=.0033;
q1=Kinfty*  (U.^0.8);
Haux= H.^0.2;
q2=q1 ./ Haux ;
q3=q2 ./ R;
q= q3.*( Cinfty - C);

function v_prime=CalDerivative( v )

[D,grid]=DerivativeSpectralCheb(length(v) -1);

v_prime=D*v';
v_prime=v_prime';
```

6.1 Additional Codes

Codes for trying the derivatives

```
function Error=TryDerivativeFourier

Error=zeros(3,1);
x=2^(-8):2^(-8):1;

f1=exp( -x.^2 ) ;
f1prime= -2 *x .* f1;
f2=exp ( -x ).* sin (2*x);
f2prime= -f2+ 2*exp ( -x ).*cos (2*x);
f3=ones( 1,length(x));
f3prime=zeros(1,length(x));
plot(x,f1prime,x,Derivative_Fourier(f1));
```

```
Error(1,1)=norm(Derivative_Fourier(f1)-f1prime,inf);
Error(2,1)=norm(Derivative_Fourier(f2)-f2prime,inf);
Error(3,1)=norm(Derivative_Fourier(f3)-f3prime,inf);

function Error=TryDerivative(N)

Error=zeros(3,N);
for i=1:N
[D,x]=DerivativeSpectralCheb(i);

f1=exp( -x.^2 ) ;
f1prime= -2 *x .* f1;
f2=exp ( -x ).* sin (2*x);
f2prime= -f2+ 2*exp ( -x ).*cos (2*x);
f3=ones( length(x),1);
f3prime=zeros(length(x),1);

Error(1,i)=norm(D*f1-f1prime,inf);
Error(2,i)=norm(D*f2-f2prime,inf);
Error(3,i)=norm(D*f3-f3prime,inf);
end
```

References

[1] Duan, J.G.,*Simulation of Flow and Mass Dispersion in Meandering Channels*, Journal of Hydraulic Engineering,(2004)

[2] Duan, J.G., Julien, *Numerical simulation of the inception of channel meandering*, Earth Surface Processes and Landforms, (2005)

[3] Duan, J.G., Julien, *Numerical Simulation of Meandering Evolution*, Journal of Hydrology,Article in Press.

[4] Gottlieb,D., Orszag, S.A.,*Numerical Analysis of Spectral Methods*, SIAM, (1977

[5] Hervouet,J-M., *Hydrodynamic of Free Surface Flows*, John Wiley & Sons, (2007)

[6] Trefethen,L.N.,*Spectral Methods in Matlab*, SIAM, (2000)

[7] Wu, W., *Computational River Dynamics*, (2007)

[8] *http://www.potash1.ca/s/AboutSolutionMining.asp*