An Introduction to PDE Constrained Optimization

Eldad Haber

Dept of Computer Science and Mathematics,
Emory University
Aug 2003

Rough Outline

- Motivation: applications
- Review of some numerical PDE techniques
- Review of optimization techniques
- Optimization and PDEs
- Some specific examples

Part 0 - Motivation: applications

Why PDE's?

Why optimization?

Some applications

Why PDE's?

PDE's describe many physical and biological phenomena fluid flow, electromagnetics, elasticity ...

Numerical solution of PDE's are used for simulations

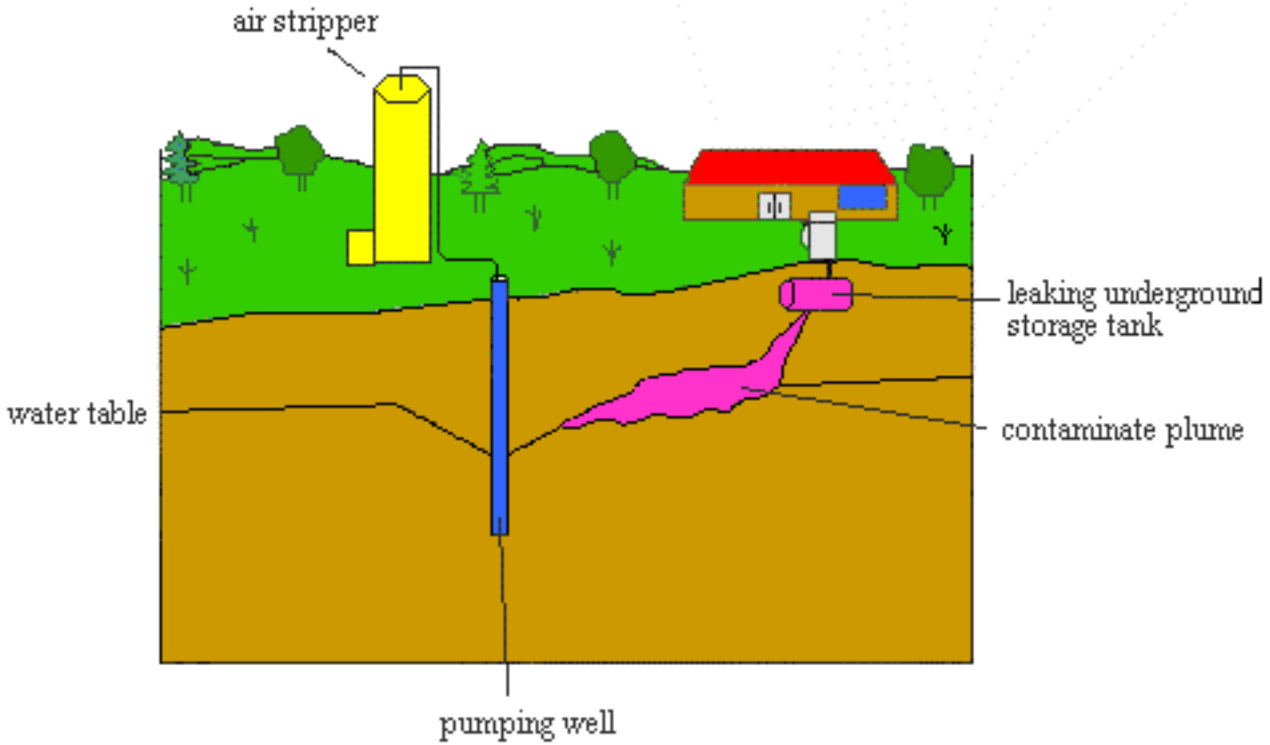
Example I - Hydrology

The pressure field is governed by the elliptic PDE

$$\nabla \cdot \sigma \boldsymbol{\nabla} u = b$$

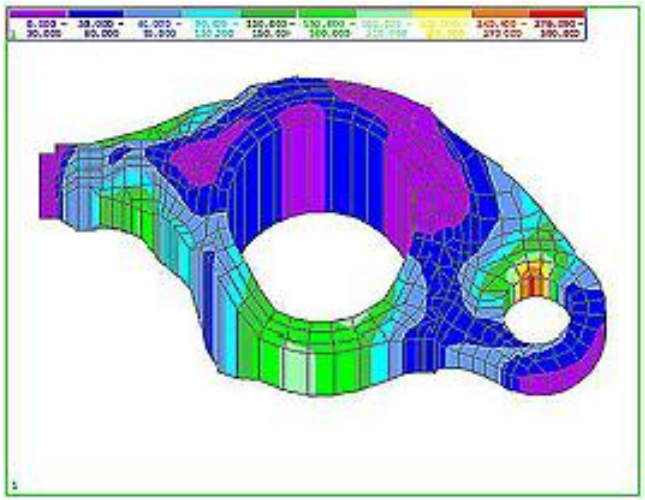
(with appropriate BC)

Simulation: Given some hydrolic conductivity σ and the source b find the pressure field u


air stripper
leaking underground
storage tank
water table
contaminate plume
pumping well

Example II - Shape modeling

Find the stress field on a mechanical part



Why Optimization

We do not want to solve **A** specific PDE

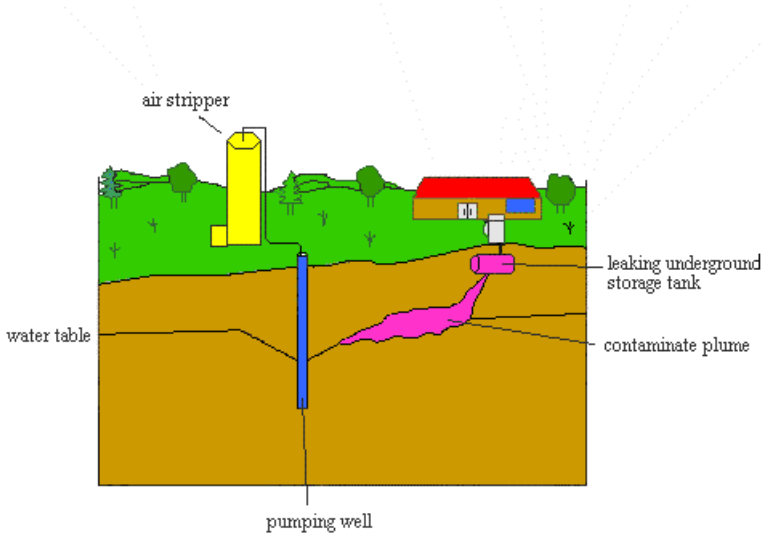
Find the “best” model

Applications:

imaging, inverse problems, optimal design ...

Optimization: Example I - Hydrology

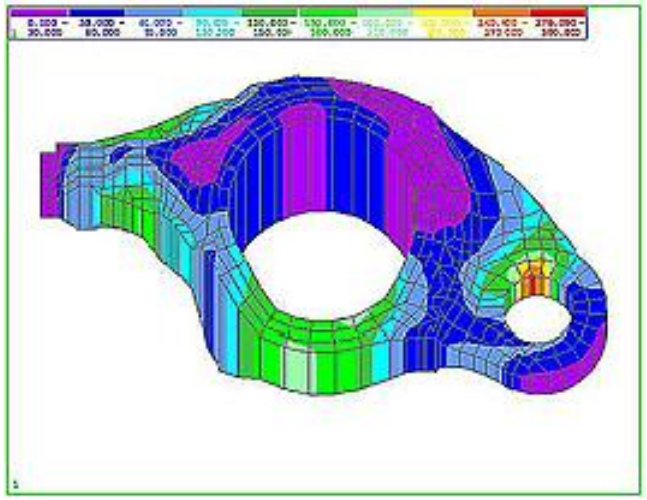
Optimization: Given some hydrolic conductivity σ and the pressure field u find the contaminating source b

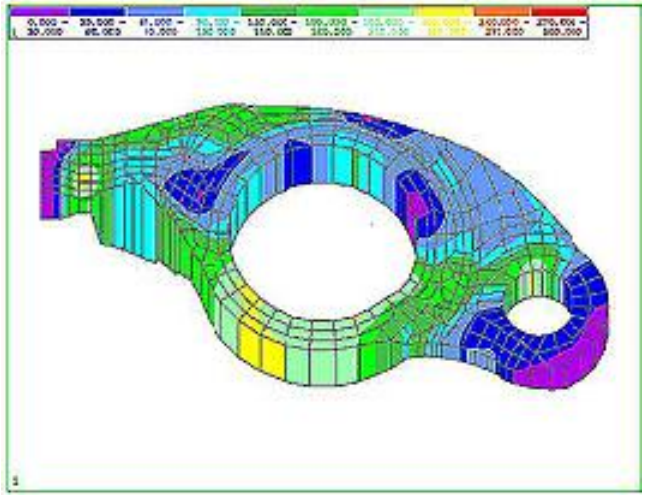


$$\min \ \frac{1}{2}\|u - u^{\text{obs}}\|^2; \quad \text{s.t} \ \ \nabla \cdot \sigma \boldsymbol{\nabla} u - b = 0$$

Optimization: Example II - Shape

Find the shape of the mechanical part with the smallest stress field





Part I - Review of numerical PDE's

- In general
- Discretization of differential operators
- Solution

In general

$$\mathcal{L}(m; u) = q$$

where $\mathcal{L}$ is a differential operator which depends on some parameters m.

Classical examples

Elliptic

$$\nabla \cdot \rho(m) \boldsymbol{\nabla} u = q; \quad u|_{\partial\Omega} = 0$$

Parabolic

$$u_t - \nabla \cdot \rho(m) \boldsymbol{\nabla} u = 0; \quad u|_{\partial\Omega} = 0; \quad u(x,0) = u_0$$

Hyperbolic

$$u_{tt} - \nabla \cdot \rho(m) \boldsymbol{\nabla} u = 0$$

$$u|_{\partial\Omega} = 0 \quad u(x,0) = u_0 \quad u_t(x,0) = v_0$$

Some things we will never agree on

- weak vs strong form
- Finite difference vs finite element
- Meshing - regular vs unstructured grid

Books: Thomas, Smith,LeVeque, Hughes, Oden, Jin ...

Differential operators and their discretization

Building blocks for many PDE's

- The gradient

$$\boldsymbol{\nabla} = \begin{pmatrix} \partial_x \\ \partial_y \end{pmatrix}$$

- The divergence

$$\nabla\cdot = \begin{pmatrix} \partial_x & \partial_y \end{pmatrix}$$

- The mass matrix

$$\sigma(\cdot)$$

Discretization using finite difference - finite volume

In 1D ∇= d/dx;

Assume we have the grid $0 < h < 2h < ... < (n-1)h < 1$

Then

$$v(jh) = \frac{du}{dx}|_{jh} = \frac{u((j+1)h) - u(jh)}{h} + \mathcal{O}(h) = \frac{u(jh) - u((j-1)h)}{h} + \mathcal{O}(h)$$
$$= \frac{u((j+1)h) - u((j-1)h)}{h} + \mathcal{O}(h^2)$$

Central difference is more accurate

But:

Central difference has a non-trivial null-space

Can generate stability problems

Solution - Use staggered grids

$$v((j+\frac{1}{2})h) = \frac{du}{dx}|_{(j+\frac{1}{2})h} = \frac{u((j+1)h)-u(jh)}{h} + \mathcal{O}(h^2)$$

$$w(jh) = \frac{dv}{dx}|_{jh} = \frac{v((j+1/2)h)-v((j-\frac{1}{2})h)}{h} + \mathcal{O}(h^2)$$

The discrete derivative operator

Two difference operators

$$D_1 : \quad \text{primal grid} \rightarrow \text{dual grid}$$

$$D_2 : \quad \text{dual grid} \rightarrow \text{primal grid}$$

$$\frac{d}{dx}_{\text{primal}} \approx D_1 = \begin{pmatrix} -1 & 1 & & & \\ & -1 & 1 & & \\ & & -1 & 1 & \\ & & & -1 & 1 \end{pmatrix}$$

Easy to show

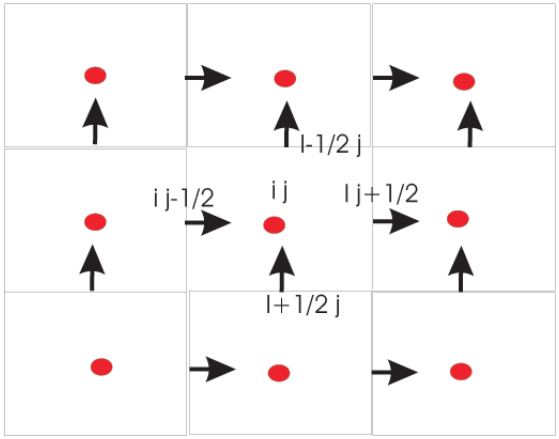
$$D_2 = -D_1^T = -D^T$$

Properties of the discrete derivative operator

Our staggered grid difference operator mimics the continuous derivative operator

$$\begin{aligned} \text{inner product} \qquad & (Du, v) = -(u, D^* v) \\ \text{null space} \qquad & De = 0 \end{aligned}$$

Derivative operator in 2D: The gradient



$$\mathbf{J}_x = u_x \approx D_x u$$
$$\mathbf{J}_y = u_y \approx D_y u$$

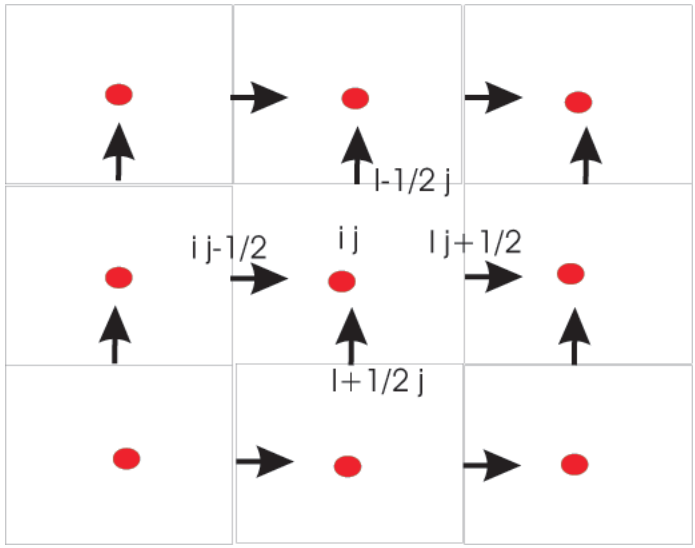
Derivative operator in 2D: The gradient

$$D_x = \begin{pmatrix} -1 & 1 & & & & & & & & \\ & -1 & 1 & & & & & & & \\ & & -1 & 1 & & & & & & \\ & & & -1 & 1 & & & & & \\ & & & & & -1 & 1 & & & \\ & & & & & & -1 & 1 & & \\ & & & & & & & -1 & 1 & \\ & & & & & & & & -1 & 1 \end{pmatrix}$$

$$D_y = \begin{pmatrix} -1 & & & & & 1 & & & & \\ & -1 & & & & & 1 & & & \\ & & -1 & & & & & 1 & & \\ & & & -1 & & & & & 1 & \end{pmatrix}$$

$$\nabla_h = \begin{pmatrix} D_x \\ D_y \end{pmatrix}$$

Derivative operator in 2D: The div



Define - mass-balance

$$\nabla_h \cdot \mathbf{J} = h^{-1}\left((J_x^+ - J_x^-) + (J_y^+ - J_y^-)\right)$$

Properties of the div-grad

Mimetic properties

$$\nabla_h \cdot = -\boldsymbol{\nabla}_h^T$$

$$\boldsymbol{\nabla}_h : \text{cell center} \rightarrow \text{faces}$$

$$\nabla_h \cdot : \text{faces} \rightarrow \text{cell centers}$$

$$\text{inner product} \qquad (\nabla_h \cdot \mathbf{J}, u) = -(\mathbf{J}, \boldsymbol{\nabla}_h u)$$

$$\text{null space} \qquad \boldsymbol{\nabla}_h e = 0$$

The mass matrix

Matrix represents $\sigma(\cdot)$

Note - If $\sigma = 1$ should look like identity

Where does the mass matrix appear?

Break $\nabla \cdot \sigma \boldsymbol{\nabla} u = q$ into first order

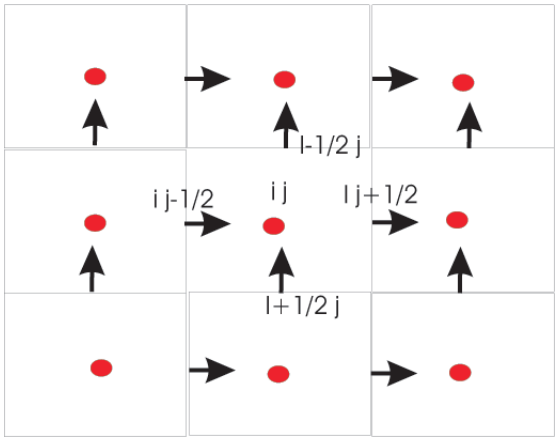
$$\nabla \cdot \mathbf{J} = q$$

$$\sigma^{-1}\mathbf{J} - \boldsymbol{\nabla} u = 0$$

The mass matrix

The discrete system

$$\nabla_h \cdot \mathbf{J} = q$$
$$\sigma_h^{-1}\mathbf{J} - \boldsymbol{\nabla}_h u = 0$$



σ_h^{-1} is the value of the conductivity on cell faces

May need averaging ...

Put it all together I

We have introduced mimetic discretization for the operators $\nabla\cdot, \boldsymbol{\nabla}, \sigma$

We can now build

- Laplacian $\Delta = \nabla \cdot \boldsymbol{\nabla} \quad \leftrightarrow \quad \Delta_h = \nabla_h \cdot \boldsymbol{\nabla}_h$
  five point discrete Laplacian

$$h^{-2} \begin{pmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{pmatrix}$$

- div-grad $\nabla \cdot \sigma \boldsymbol{\nabla} \quad \leftrightarrow \quad \nabla_h \cdot S_h \boldsymbol{\nabla}_h$

Put it all together II

Elliptic PDE: $\nabla_h \cdot \ S_h(m)\boldsymbol{\nabla}_h u = A(m)u = q$

For Parabolic and hyperbolic get the ODE's

Parabolic PDE:

$$\nabla_h \cdot \ S_h(m)\boldsymbol{\nabla}_h u = A(m)u = u_t; \ \ u(0) = u_0$$

Hyperbolic PDE:

$$\nabla_h \cdot \ S_h(m)\boldsymbol{\nabla}_h u = A(m)u = u_{tt}; \ \ u(0) = u_0; \ \ u'(0) = v_0$$

Discretization in time

Method of lines - treat as ODE's

For stiff system (heat) use implicit (say midpoint)

$$\Delta t^{-1}(u_{n+1} - u_n) = \frac{1}{2}A(m)(u_{n+1} + u_n)$$

For hyperbolic system use explicit (say leap-frog)

$$\Delta t^{-2}(u_{n+1} - 2u_n + u_{n-1}) = A(m)u_n$$

Implicit vs Explicit

Issues to think of

In general:

- Imp are unconditionally stable while Exp are not.
- Imp involve matrix inversion while Exp not

Typical stability for Exp method (CFL)

$$c\Delta t = \Delta x \quad (\text{hyperbolic})$$
$$c\Delta t = \Delta x^2 \quad (\text{parabolic})$$

Discretization in time

In the implicit case, the problem in time for all times can be written as

$$\widehat{A}(m) = \begin{pmatrix} I - \frac{1}{2}A(m) & & & \\ \frac{1}{2}A - I & I - \frac{1}{2}A(m) & & \\ & \ddots & \ddots & \\ & & \frac{1}{2}A - I & I - \frac{1}{2}A(m) \end{pmatrix}$$

Issues for time discretization

Implicit methods are unconditionally stable

Explicit methods are not!

Stability of an explicit method depends on the parameter m, the time step Δt and the spacing Δx

(Return to this point later)

A note on boundary conditions

Rule of thumb: 90% of code development is in the Boundary Conditions.

May be nontrivial but very important in order to get the correct results.

Could alter the equations (e.g. PML for hyperbolic systems)

Properties of many matrices that evolve from PDE's

Diagonally dominant (no convection)

Eigenvalues cluster at infinity

Problem becomes more ill-conditioned as grid refined

Solution techniques - briefly

Krylov-Krylov-Krylov (MG) (See short course by Ramage)

But

Usually too slow without a decent preconditioner.

If possible, use the fact that the matrix is diagonally dominant

$$M^{-1}Au = M^{-1}q$$

Part II - Review of numerical optimization

- In general
- Unconstrained optimization
- Constrained optimization

In general

$$\min_{\mathbf{m}\in\Omega} \quad f(\mathbf{m})$$

$$\Omega = \{\mathbf{m} \in \mathbb{R}^n | c_i(\mathbf{m}) = 0,\ i \in \mathcal{E},\quad c_i(\mathbf{m}) \geq 0,\ i \in \mathcal{I}\}$$

Here we will consider only equality constraints

Some things we never agree on

- continuous vs discrete optimization
- global vs local optimization
- stochastic vs deterministic optimization
- unconstrained and constrained optimization

Books: Nocedal-Wright; Fletcher; Gill, Murray & Wright; Luenberger, ...

Unconstrained optimization

$$\min_{\mathbf{m}\in\mathbb{R}^n} \quad f(\mathbf{m})$$

Look for an isolated local minimizer $\mathbf{m}^*$

First order necessary conditions: $\mathbf{m}^*$ must be a stationary point

$$\boldsymbol{\nabla} f(\mathbf{m}^*) = \mathbf{0} \quad (\text{gradient})$$

Second order necessary conditions: that the Hessian $\nabla^2 f(\mathbf{m}^*)$ be positive semi-definite; sufficient if positive definite.

Numerical algorithm

Starting from an initial iterate $\mathbf{m}_0$, generate iterates $\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_k, \ldots$ such that

$$f(\mathbf{m}_k) > f(\mathbf{m}_{k+1}) \quad (\text{sufficiently})$$

At $\mathbf{m}_k$, generate a direction $\mathbf{p} = \mathbf{p}_k$ and determine a step-size $\alpha = \alpha_k$ s.t.

$$\mathbf{m}_{k+1} = \mathbf{m}_k + \alpha_k \mathbf{p}_k.$$

Strategies

Line search: First determine a descent direction $\mathbf{p}_k$ s.t.

$$\mathbf{p}_k^T \boldsymbol{\nabla} f(\mathbf{m}_k) < 0.$$

Then determine α_k to *approximately* solve

$$\min_{\alpha} f(\mathbf{m}_k + \alpha \mathbf{p}_k).$$

Trust region: Determine $\mathbf{p}$ and control its size simultaneously

$$\min_{\mathbf{p}} \quad M_k(\mathbf{m}_k + \mathbf{p}) \quad s.t. \ \|\mathbf{p}\| \leq \Delta$$

$$M_k(\mathbf{m}_k + \mathbf{p}) = f_k + \mathbf{p}^T \boldsymbol{\nabla} f_k + \frac{1}{2} \mathbf{p}^T B_k \mathbf{p}$$

Descent directions:

- $\mathbf{p}_k = -\boldsymbol{\nabla} f_k$ Steepest descent
- $\mathbf{p}_k = -[\nabla^2 f_k]^{-1} \boldsymbol{\nabla} f_k$ Newton
- $\mathbf{p}_k = -B_k^{-1} \boldsymbol{\nabla} f_k;$ B_k any positive definite matrix

Quasi-Newton: Maintain positive definite B_k, update at each step, in place of true Hessian.

Large scale problems

Use CG for $H_k \mathbf{p} = -\nabla f_k$ (inner iteration; often requires preconditioning). Yields Newton-CG etc.

Inexact Newton-type

Terminate iterative procedure for inner iteration before convergence.

Typically must require at least

$$\begin{aligned} \|\mathbf{r}_k\| &\leq \eta_k \|\nabla f_k\| \quad 0 \leq \eta_k \leq \eta < 1 \\ \mathbf{r}_k &= B_k \mathbf{p}_k + \nabla f_k \end{aligned}$$

(for faster convergence may need $\eta_k \to 0$ or $\eta_k = O(\|\nabla f_k\|$)

Quasi-Newton methods

Replace $H_k \mathbf{p} = -\nabla f_k$ with

$$B_k \mathbf{p} = -\nabla f_k; \qquad B_k = \sum_{i=1}^{L} \rho_i u_i u_i^T$$

Most successful: L-BFGS

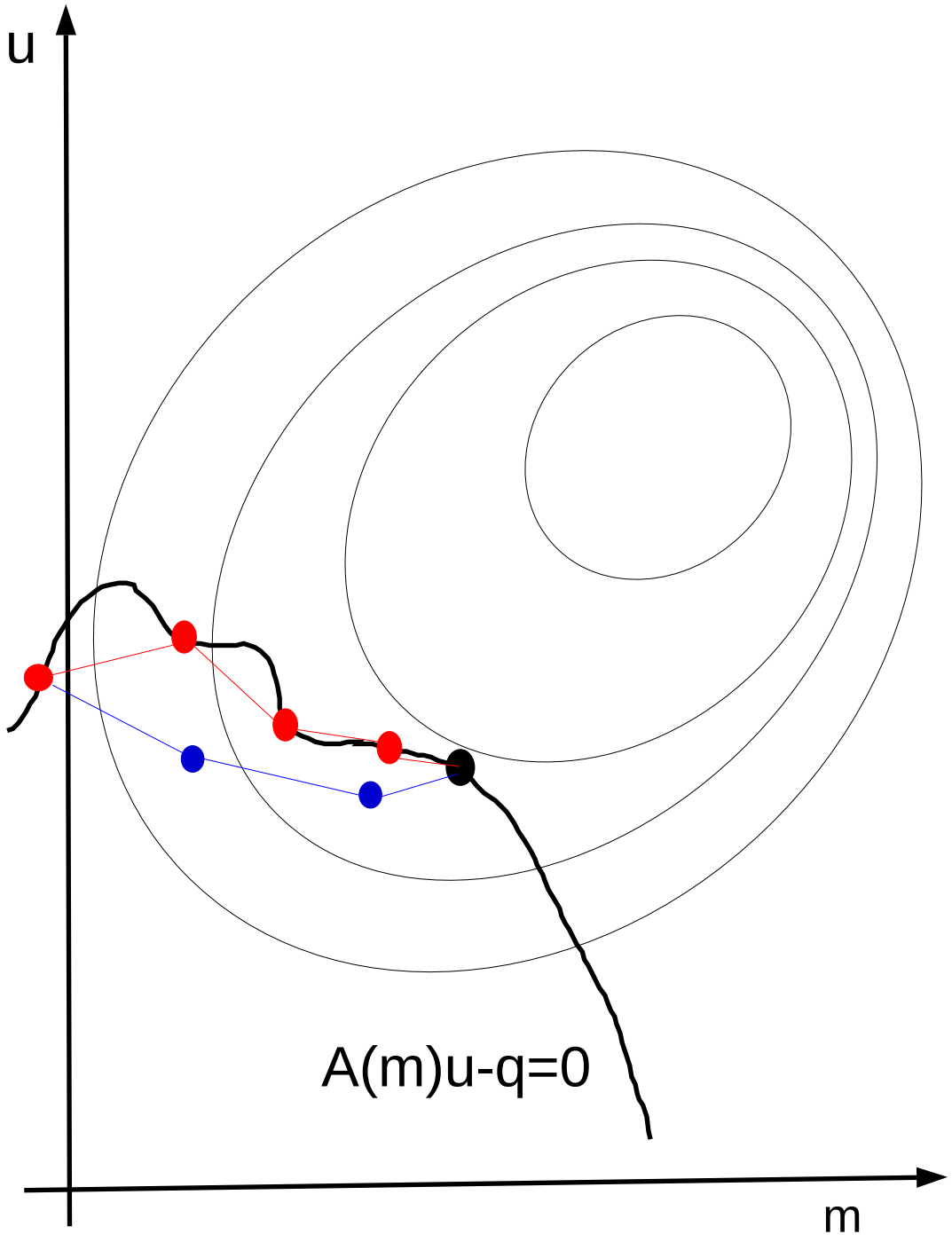
- Easy to solve $B_k \mathbf{p} = -\nabla f_k$
- Can store only a limited number of vectors

Equality Constrained optimization

$$\min_{\mathbf{m}\in\Omega} \quad f(\mathbf{m};u)$$
$$\text{s.t} \;\; C(\mathbf{m};,u)=0$$

Lagrangian

$$\mathcal{L}(\mathbf{m},u,\boldsymbol{\lambda}) = f(\mathbf{m};u) - \sum_i \lambda_i C_i(\mathbf{m};u)$$


u
m
A(m)u-q=0

Constraint qualification: Assume

$$\{B = [C_{\mathbf{m}}(\mathbf{m}^*; u^*), C_u(\mathbf{m}^*; u^*)],)\}$$

is linearly independent (i.e. the matrix B^T has full column rank).

First order necessary conditions (KKT):

$$\begin{aligned} \boldsymbol{\nabla}_{\mathbf{m}} f(\mathbf{m}^*, u^*) + C_{\mathbf{m}}^T \boldsymbol{\lambda}^* &= 0 \\ \boldsymbol{\nabla}_u f(\mathbf{m}^*, u^*) + C_u^T \boldsymbol{\lambda}^* &= 0 \\ C(\mathbf{m}^*; u^*) &= 0 \end{aligned}$$

Approaches and methods

Constraint elimination

$$u = G(\mathbf{m}) \quad \leftrightarrow \quad C(\mathbf{m}; u) = 0$$
$$\min \quad f(\mathbf{m}; G(\mathbf{m}))$$

Penalty

$$\min f(\mathbf{m}; u) + \frac{1}{2\mu} \sum_i C_i^2(\mathbf{m}; u), \quad \mu > 0$$

$$\min f(\mathbf{m}; u) + \frac{1}{\mu} \sum_i |C_i(\mathbf{m}; u)|, \quad \mu > 0$$

Augmented Lagrangian

$$\min \mathcal{L}_A = f(\mathbf{m}) - \sum_i \lambda_i C_i(\mathbf{m}; u) + \frac{1}{2\mu} \sum_i C_i^2(\mathbf{m}; u), \quad \mu > 0$$

Sequential Quadratic Programming (SQP)

$$\begin{aligned} \min_{p_{\mathbf{m}}, p_u} \quad & f(\mathbf{m}, u) + \begin{pmatrix} \nabla_{\mathbf{m}} f_k^T & \nabla_u f_k^T \end{pmatrix} \begin{pmatrix} p_{\mathbf{m}} \\ p_u \end{pmatrix} + \frac{1}{2} \begin{pmatrix} p_{\mathbf{m}}^T & p_u^T \end{pmatrix} W_k \begin{pmatrix} p_{\mathbf{m}} \\ p_u \end{pmatrix} \\ s.t. \quad & C_{\mathbf{m}} p_{\mathbf{m}} + C_u p_u + \mathbf{c}_k = \mathbf{0} \end{aligned}$$

Objective approximates Lagrangian ($W_k \approx \nabla^2 \mathcal{L}$).

Line search using merit function weighing objective and insfeasibilities, e.g.,

$$f(\mathbf{m}; u) + \frac{1}{\mu} \sum_i |C_i(\mathbf{m}; u)|$$

Linear systems in SQP

$$\begin{pmatrix} W_k & B_k^T \\ B_k & 0 \end{pmatrix} \begin{pmatrix} \mathbf{p} \\ \lambda \end{pmatrix} = - \begin{pmatrix} f_k \\ \mathbf{c}_k \end{pmatrix}$$

SQP - Solve and update

Difficulties

B_k can be very large (PDE)

The system can be strongly coupled

W_k can be singular

The whole KKT system can be ill-conditioned

Part III - PDE's and optimization

- What is special about PDE's and optimization?
- How should we solve such problems?
  - The PDE side
  - The optimization side
  - Combined view

PDE's and optimization

Huge number of constraints (even for "small" problems)

Conditioning may be very bad

Usually - the "meat" of the problem is on the PDE side.

Model problem

Image processing, hydrology, impedance tomography ...

$$\begin{aligned} \min \quad & \phi(m;u) = \frac{1}{2}\|Qu - b\|^2 + \beta R(m) \\ \text{s.t} \quad & \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} u - q = 0 \end{aligned}$$

Popular choices for $R(m)$

TV - $R(m) = \int \sqrt{|\boldsymbol{\nabla} m|^2 + \gamma^2}\, dV$

L2 - $R(m) = \frac{1}{2} \int (\boldsymbol{\nabla} m)^T (\boldsymbol{\nabla} m)\, dV$

Model problem - discretize then optimize

- Discretize the optimization problem first (grid h)
- Solve a discrete optimization problem

$$\begin{aligned} \min \quad & \phi(m;u) = \frac{1}{2}\|u-b\|^2 + \frac{1}{2}\beta h^d (\boldsymbol{\nabla}_h m)^T (\boldsymbol{\nabla}_h m) \\ \text{s.t} \quad & \nabla_h \cdot M(m) \boldsymbol{\nabla}_h u = A(m)u = q \end{aligned}$$

Model problem - Constraint elimination

We can eliminate $u = A(m)^{-1}q$

Obtain

$$\min \ \phi(m) = \frac{1}{2}\|A(m)^{-1}q - b\|^2 + \frac{1}{2}\beta h^d (\boldsymbol{\nabla}_h m)^T (\boldsymbol{\nabla}_h m)$$

- Unconstrained optimization
- Function evaluation involve solving a (large) linear system
- Gradient hard to evaluate

Model problem - The Constrained approach

Do not eliminate the constraints

$$\mathcal{L}(m,u,\lambda) = \frac{1}{2}\|u-b\|^2 + \frac{1}{2}\beta h^d (\boldsymbol{\nabla}_h m)^T (\boldsymbol{\nabla}_h m) + \lambda^T (A(m)u - q)$$

Gradient

$$\begin{aligned}
\mathcal{L}_u &= u - b + A(m)^T \lambda = 0 \\
\mathcal{L}_m &= \beta h^d (\boldsymbol{\nabla}_h^T \boldsymbol{\nabla}_h) m + \left(\frac{\partial (A(m)u)}{\partial m}\right)^T \lambda = 0 \\
\mathcal{L}_\lambda &= A(m)u - q = 0
\end{aligned}$$

Model problem - The PDE view

A discretization of the coupled nonlinear PDE system

$$
\begin{aligned}
& u - b + A(m)^T \lambda = 0 \\
& \beta(\boldsymbol{\nabla}_h^T \boldsymbol{\nabla}_h) m + G(m,u)^T \lambda = 0 \\
& A(m)u - q = 0
\end{aligned}
\qquad\qquad
\begin{aligned}
& u - b + \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} \lambda = 0 \\
& -\beta \Delta m + (\boldsymbol{\nabla} u)^T \mathrm{e}^m (\boldsymbol{\nabla} \lambda) = 0 \\
& \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} u - q = 0
\end{aligned}
$$

$$
G(m,u) = \left(\frac{\partial (A(m)u)}{\partial m} \right)
$$

Model problem - Newton's method - KKT system

$$\begin{pmatrix} I & & A^T \\ & \beta(\boldsymbol{\nabla}_h^T \boldsymbol{\nabla}_h) & G^T \\ A & G & 0 \end{pmatrix} \begin{pmatrix} s_u \\ s_m \\ s_\lambda \end{pmatrix} = \text{rhs}$$

$$\begin{pmatrix} I & & \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} \\ & -\beta\Delta & (\boldsymbol{\nabla} u)^T \mathrm{e}^m \boldsymbol{\nabla} \\ \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} & \nabla \cdot ((\boldsymbol{\nabla} u) \mathrm{e}^m(\cdot)) & 0 \end{pmatrix} \begin{pmatrix} s_u \\ s_m \\ s_\lambda \end{pmatrix} = \text{rhs}$$

Solving KKT systems I

Method I - If β large reorder as a block diagonal dominant. [Precondition by the diagonal blocks]

$$\begin{pmatrix} A & 0 & G \\ I & A^T & \\ & G^T & \beta(\boldsymbol{\nabla}_h^T \boldsymbol{\nabla}_h) \end{pmatrix} \begin{pmatrix} s_u \\ s_\lambda \\ s_m \end{pmatrix} = \mathrm{rhs}$$

$$\begin{pmatrix} \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} & & \nabla \cdot ((\boldsymbol{\nabla} u)\mathrm{e}^m(\cdot)) \\ I & \nabla \cdot \mathrm{e}^m \boldsymbol{\nabla} & \\ & (\boldsymbol{\nabla} u)^T \mathrm{e}^m \boldsymbol{\nabla} & -\beta\Delta \end{pmatrix} \begin{pmatrix} s_u \\ s_\lambda \\ s_m \end{pmatrix} = \mathrm{rhs}$$

Solving KKT systems II

Method II-III - If β small coupled systems of PDE's

- Multigrid (for the coupled system)
- Preconditioners based on elimination (optimization based)

Easy to note that the KKT matrix can be decomposed to

$$\begin{pmatrix} A & 0 & G \\ I & A^T & 0 \\ 0 & G^T & \beta R'' \end{pmatrix}^{-1} =$$

$$\begin{pmatrix} A^{-1} & 0 & -A^{-1}GH_r^{-1} \\ 0 & A^{-T} & -A^{-T}JH_r^{-1} \\ 0 & 0 & H_r^{-1} \end{pmatrix} \cdot \begin{pmatrix} I & 0 & 0 \\ A^{-1} & I & 0 \\ -J^TA^{-1} & -G^TA^{-T} & I \end{pmatrix}$$

$$J = A^{-1}G$$

$$H_r = J^TJ + \beta\boldsymbol{\nabla}_h^T\boldsymbol{\nabla}_h$$

Solving KKT systems II

Preconditioners - Use PDE techniques to approximate A^{-1} (multigrid, ILU ...)

Combine to approximate the decomposition of the KKT matrix

Use algebraic techniques (elimination, Schur complement) if needed

Solving KKT systems - recap

- Solving KKT systems may not be hard if they correspond to a sparse decoupled PDE

- We should remember the dual view matrix-PDE

Globalization

Update

$$m \leftarrow m + \alpha s_m; \quad u \leftarrow u + \alpha s_u; \quad \lambda \leftarrow \lambda + \alpha s_\lambda;$$

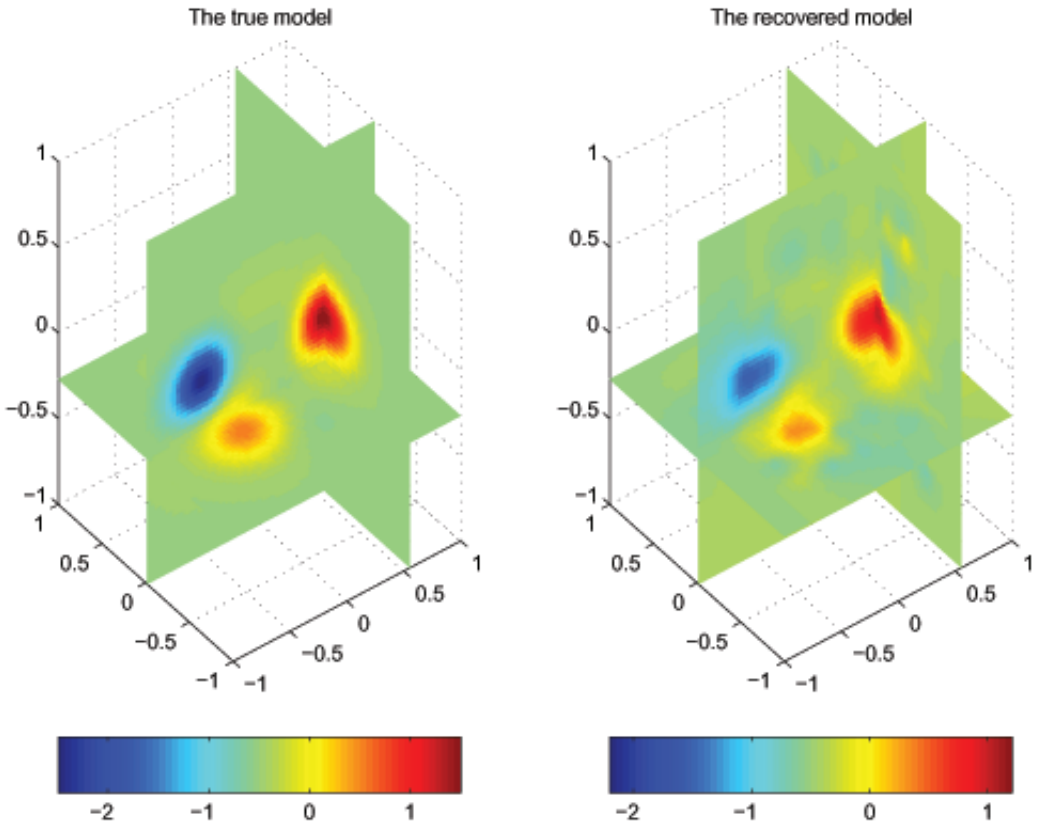
Line search using merit function weighing objective and insfeasibilities, e.g.,

$$\frac{1}{2}(u-b)^2 + \beta R(m) + \frac{1}{\mu}\|A(\mathbf{m})u - q\|_1$$

Example I

Discretize on a 64^3 grid

$\beta = 1e-1$	misfit = 0.1		
nonlin it	KKT it	constr	rel-grad
1	2	$3e-3$	1e-2
2	3	$2e-4$	4e-3
3	2	$7e-6$	1e-3
4	2	$9e-7$	3e-4
$\beta = 1e-2$	misfit = 0.04		
Nonlin it	KKT it	const	grad
1	7	$4e-6$	2e-3
2	5	$6e-7$	7e-4


The true model
The recovered model
1
0.5
0
−0.5
−1
−2
−1
0
1

Model problem II - some pitfalls

Wave propagation in 1D

$$\begin{aligned} \min \quad & \phi(m;u) = \frac{1}{2}\|Qu-b\|^2 + \beta\|\boldsymbol{\nabla} m\|^2 \\ \text{s.t} \quad & mu_{xx} - u_{tt} = 0; \quad u(0) = q_0; \quad u_t(0) = 0 \end{aligned}$$

Model problem II

For simplicity here use leapfrog in time

$$B(m)u_n - (u_{n+1} - 2u_n + u_{n-1}) = 0$$

Important Stability condition (CFL): $m^{-1}\Delta t < \Delta x$

In matrix notation

$$A(m)u = \begin{pmatrix} I & & & & & & \\ & I & & & & & \\ -I & B+2I & -I & & & & \\ & -I & B+2I & -I & & & \\ & & -I & B+2I & -I & & \\ & & & -I & B+2I & -I & \\ & & & & -I & B+2I & -I \end{pmatrix} \begin{pmatrix} u_0 \\ u_1 \\ . \\ . \\ . \\ u_n \end{pmatrix} = \begin{pmatrix} q_0 \\ q_0 \\ 0 \\ . \\ . \\ . \end{pmatrix} = q$$

Note - to solve, no need for matrix inversion

Model problem II

The problem has the same form as before

$$\begin{aligned} \min \quad & \phi(m;u) = \frac{1}{2}\|Qu-b\|^2 + \beta\|\boldsymbol{\nabla} m\|^2 \\ \text{s.t} \quad & A(m)u = q \end{aligned}$$

Try to use the same optimization techniques ...

Model problem II - numerical experiment

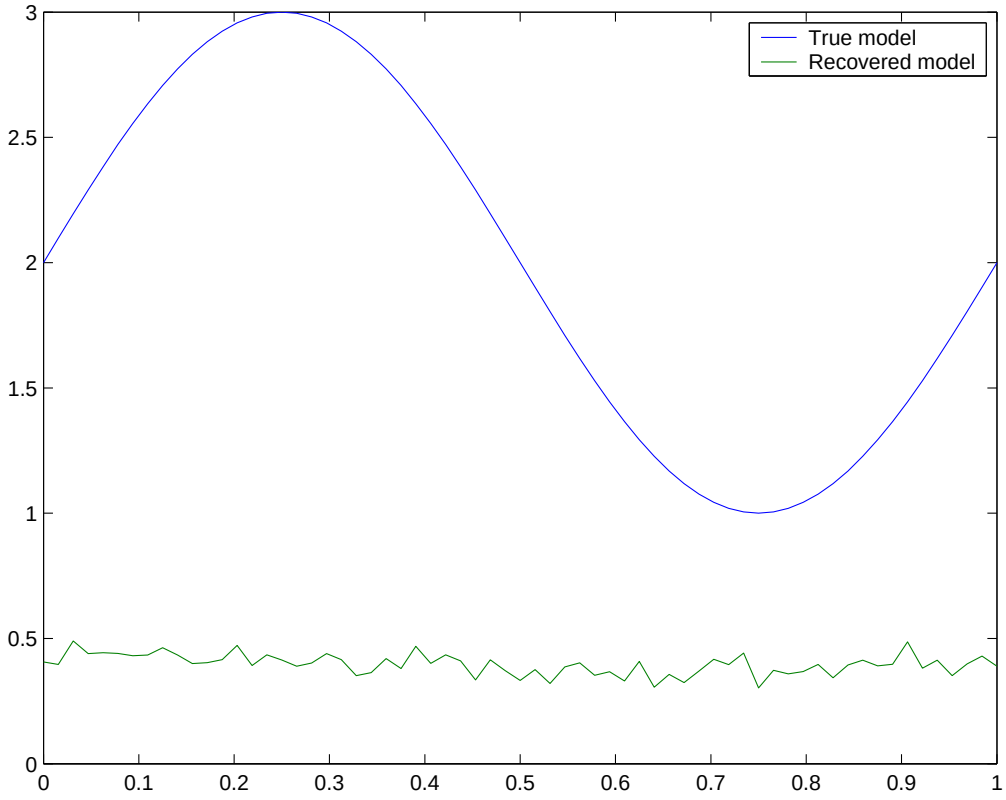
Numerical experiments

$\Delta x = 1.56\mathrm{E}-2$

$\Delta t = 3.12\mathrm{E}-2$

Initial $m^{-1} = \mathrm{const} = 0.2$

CFL condition - $0.2 \times \Delta x > \Delta t = 3.12\mathrm{E}-2$


True model
Recovered model
3
2.5
2
1.5
1
0.5
0
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Model problem II - numerical experiment - What went wrong

For the true m CFL is

$$m_{\text{true}} \times \Delta x > \Delta t$$

The true m generates instabilities!

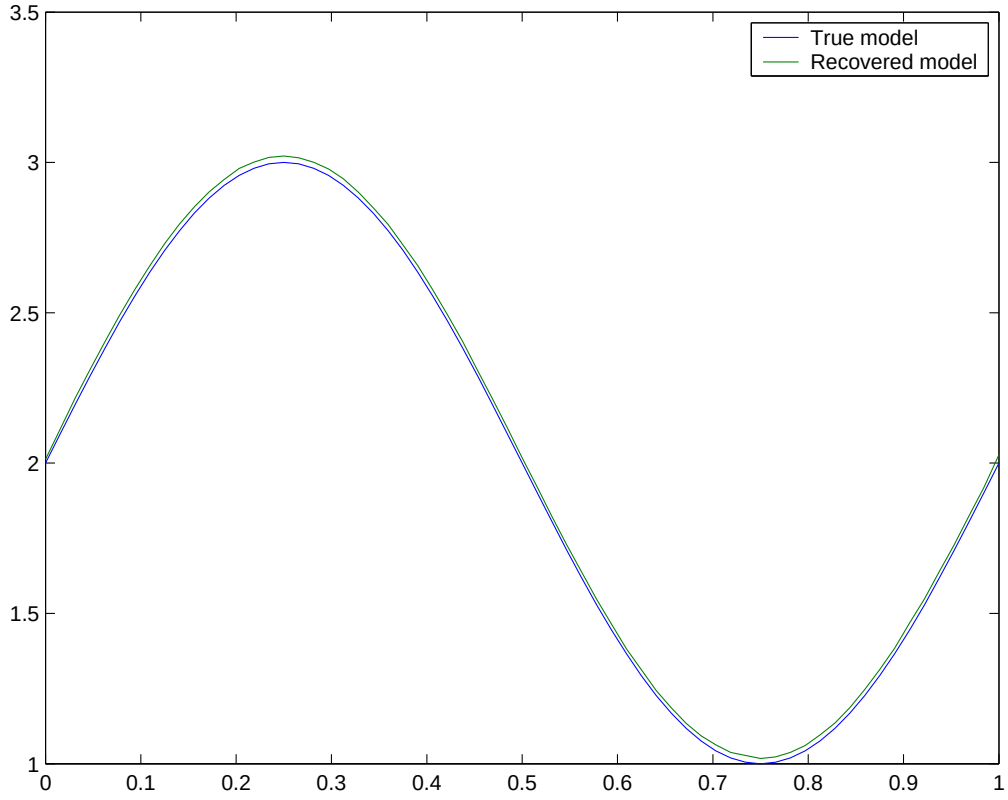
Model problem II - numerical experiment II

Numerical experiments

$\Delta x = 1.56\mathrm{E}-2$

$\Delta t = 5.0\mathrm{E}-3$

Initial $m^{-1} = \mathrm{const} = 0.2$


True model
Recovered model
3.5
3
2.5
2
1.5
1
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

Model problem II - Recap

- Discretize and then optimize yield a solvable optimization problem
- The solution of the discrete problem had nothing to do with the solution of the continuous problem
- Need to think about the PDE when doing the optimization

Summary -PDE's and optimization

- PDE's and optimization are used for many problems
- To use optimization within a PDE framework need simplicity
- Think dual
- Use best of two worlds
- Many many open questions