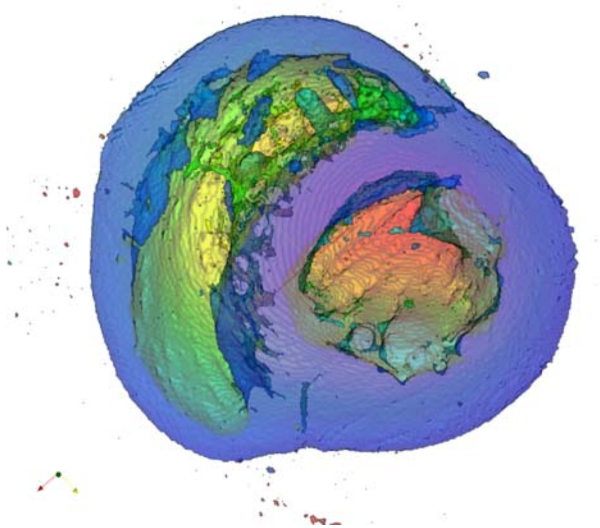
# Fast Algorithms for Inverse problems for systems constrained by reaction-diffusion equations



$$J(y,v,u) = \min_{y,v,u} \frac{1}{2}\int_T\int_{\partial\Omega} (y-y^*)^2 \, \mathrm{d}\Omega \, \mathrm{d}T + \frac{\beta}{2}\int_\Omega u^2 \, \mathrm{d}\Omega,$$

Subject to

$$\frac{\partial y}{\partial t} = D\Delta y + y(y-u)(1-y) - v,$$

$$\frac{\partial v}{\partial t} = by - \gamma v,$$

$$\nabla y(\Gamma,t).n = 0, \quad y(\Omega,0) = 0, \quad v(\Omega,0) = 0.$$

George Biros
University of Pennsylvania

Penn

# Acknowledgments

- S. Adavani, H. Sundar, S. Rahul (students)
- C. Davatzikos, D. Shen, H. Litt (imaging group)
- Department of Energy
  - DE-FG02-04ER25646
- National Science Foundation
  - CCF/ITR 0427695
- PSC, Argonne (PETSc group)

# Overview

- Motivating applications
- Inverse medium for reaction-advection-diffusion equations
- Multigrid for inverse problems
  - parabolic, elliptic
- End-to-end octree algorithms and data structures
- Fast heat potentials

# Cardiac inverse problem

Given endo- and epicardial potentials compute excitability

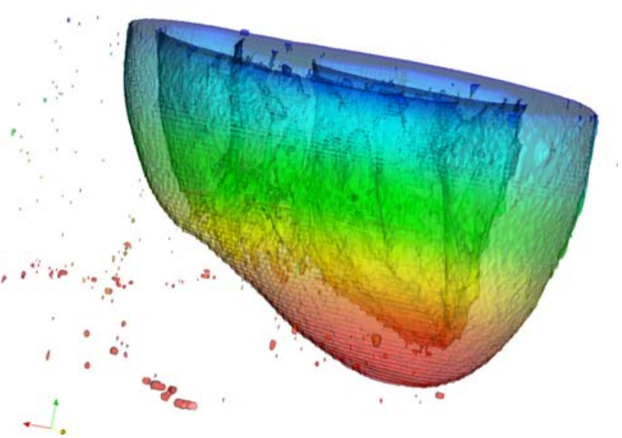
$$J(y, v, u) = \min_{y,v,u} \frac{1}{2} \int_T \int_{\partial\Omega} (y - y^*)^2 \, d\Omega \, dT + \frac{\beta}{2} \int_\Omega u^2 \, d\Omega,$$

Subject to

$$\frac{\partial y}{\partial t} = D\Delta y + y(y-u)(1-y) - v,$$

$$\frac{\partial v}{\partial t} = by - \gamma v,$$

$$\nabla y(\Gamma, t).n = 0, \quad y(\Omega, 0) = 0, \quad v(\Omega, 0) = 0.$$



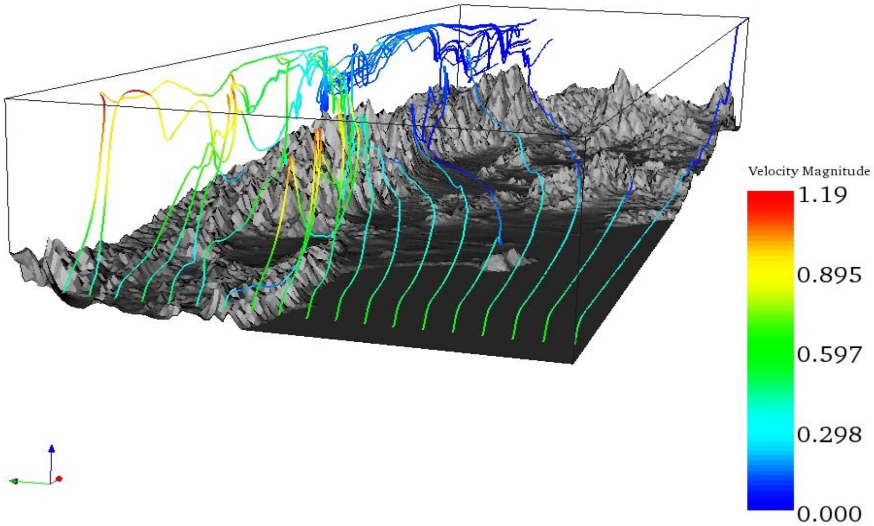
# Computational work for the forward electrophysiology problem

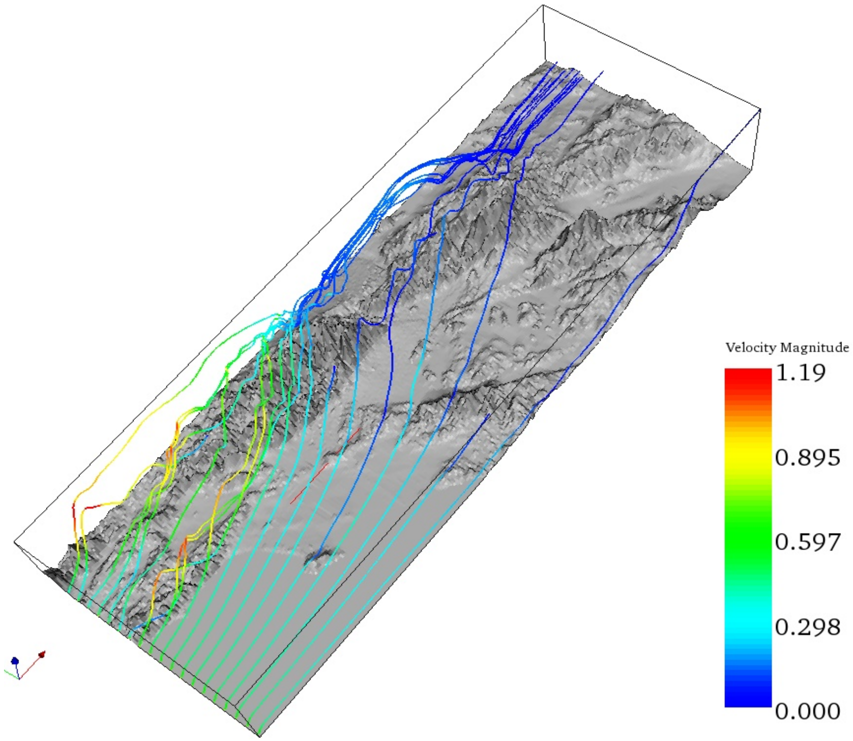
- System of reaction-diffusion PDEs
- Spatial resolution ~ 6*256^3 ~ 100 million
- Temporal resolution ~ 1000 time steps
- Inversion parameters
  - ~50-100K

- The associated costs suggest an inexact reduced space approach
  - Several other problems in this category

Penn

# A similar, "simpler" problem




- Wind from mesoscale models (MM5)
- Sparse sensor readings of concentration

Akcelic, Biros, Draganescu, Ghattas, Hill, Waanders '05

# Formulation

- Given wind *w*, observations *y**, estimate *u* :

$$
\begin{aligned}
\min_{y,u} \quad & \sum_j \int_\Omega \int_T (y - y^*)^2 \delta(\boldsymbol{x} - \boldsymbol{x}_j)\, d\boldsymbol{x}\, dt + \frac{\beta}{2} \int_\Omega u^2\, d\boldsymbol{x} \\
\text{subject to} \quad & y_t - \Delta y + \boldsymbol{w} \cdot \nabla y = 0 \quad \text{in} \quad \Omega \times (0, T) \\
& y = u \quad \text{in} \quad \Omega \times \{t = 0\} \\
& \nu \nabla y \cdot \boldsymbol{n} = 0 \quad \text{on} \quad \Gamma_N \times (0, T) \\
& y = 0 \quad \text{on} \quad \Gamma_D \times (0, T)
\end{aligned}
$$

- Then forward problem can be used to predict transport contaminant

# Need for multilevel solvers for high-resolution inverse medium problems

- Problems with single level solver
  - Algorithmic scalability
  - Globalization (for nonlinear problems)
  - CG scales well if Hessian is compact perturbation of identity
    - Multigrid to reduce constant
  - Inverse problems
    - Need solver robustness wrt regularization parameter

# Consider 1D "inverse medium" heat equation

1D parabolic PDE $\quad \Omega = (0,1),\; T = (0,1)$

$$\text{minimize } \frac{1}{2}\int_T \int_\Omega (y - y^*)^2 \, d\,\Omega \, dt + \frac{\beta}{2}\int_\Omega u^2 \, d\,\Omega$$

subject to

$$\frac{\partial y}{\partial t} - \nu \Delta y + u(x)1(t) = 0$$

$$y(x,0) = 0,\; x \in \Omega, \quad y(0,t) = y(1,t) = 0.$$

# Optimality conditions

Forward

$$\frac{\partial y}{\partial t} - \nu \Delta y + u(x)1(t) = 0$$

$$y(x,0) = 0, \; x \in \Omega, \quad y(0,t) = y(1,t) = 0$$

Adjoint

$$-\frac{\partial p}{\partial t} - \nu \Delta p + y = y^*$$

$$p(x,T) = 0, \; x \in \Omega, \quad p(0,t) = p(1,t) = 0$$

Control

$$\beta u + \int_T p \, dt = 0$$

KKT optimality conditions

---

$$\begin{bmatrix} \mathbf{I} & 0 & -\frac{\partial}{\partial t} - \Delta \\ 0 & \beta \mathbf{I} & \int_0^T dt \\ \frac{\partial}{\partial t} - \Delta & 1(t) & 0 \end{bmatrix}$$

KKT matrix

# Optimality conditions – operator form

Discretized optimality system:

$$\begin{bmatrix} \boldsymbol{O}^T\boldsymbol{O} & 0 & \boldsymbol{J}^T \\ 0 & \beta\boldsymbol{R} & -\boldsymbol{C}^T \\ \boldsymbol{J} & -\boldsymbol{C} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{u} \\ \boldsymbol{p} \end{bmatrix} = \begin{bmatrix} \boldsymbol{O}^T\boldsymbol{y}^* \\ 0 \\ 0 \end{bmatrix}$$

Reduced Hessian system:

$$(\boldsymbol{C}^T\boldsymbol{J}^{-T}\boldsymbol{O}^T\boldsymbol{O}\boldsymbol{J}^{-1}\boldsymbol{C} + \beta\boldsymbol{R})\boldsymbol{u} = -\boldsymbol{C}^T\boldsymbol{J}^{-T}\boldsymbol{O}^T\boldsymbol{y}^*$$

or

$$\boldsymbol{H}\boldsymbol{u} = \boldsymbol{g}$$

where

$$\begin{aligned} \boldsymbol{J} &: \text{ forward operator} \\ \boldsymbol{J}^T &: \text{ adjoint operator} \\ \boldsymbol{R} &: \text{ regularization operator} \\ \boldsymbol{O} &: \text{ observation operator} \\ \boldsymbol{C} &: \text{ extension of } \Omega \text{ into } \Omega \times (0,T) \end{aligned}$$

Penn

# Spectrum of reduced Hessian in the unit box with Dirichlet BCs

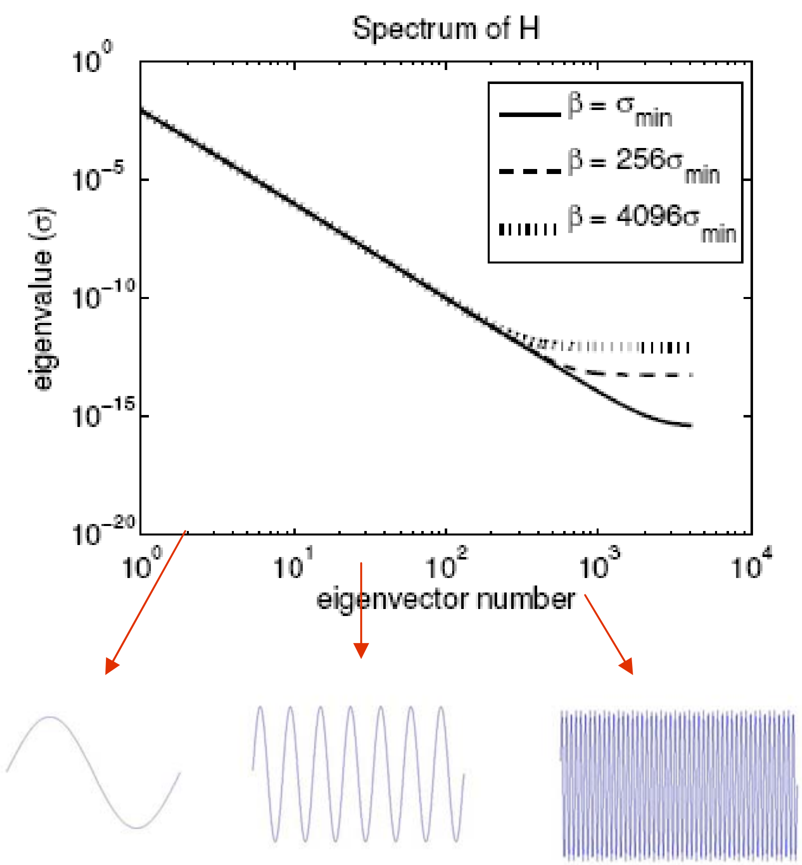
- Green's function
  $G(x, y; t) := \sum_{k=1}^{\infty} e^{-\lambda_k t}\ 2 \sin(k\pi x) \sin(k\pi y)$

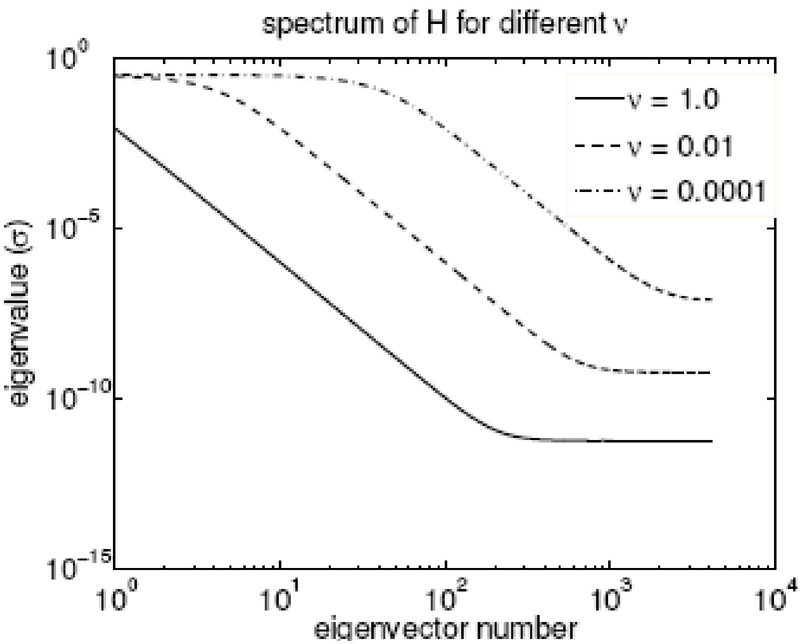
- Reduced Hessian operator
  $Hu :=$
  $\beta u(x) + \int_T 1(t) \int_T \int_\Omega \int_T \int_\Omega G(x, y; T-t-\tau) G(y, z; \tau-\sigma) 1(\sigma) u(z)\, d\Omega\, d\sigma\, d\Omega\, d\tau\, dt$

- Laplacian
  $\lambda_k = \nu k^2 \pi^2$

- Reduced Hessian
  $h_k = \beta + \frac{3 + e^{-2\lambda_k T} - 4e^{-\lambda_k T} + 2\lambda_k T}{2\lambda_k^3} = \beta + \mathcal{O}(\frac{1}{\lambda_k^2})$

# Reduced Hessian spectrum




*Use CG as a solver?*

# Using CG for the reduced Hessian

regularization

| $N_s$ | $\beta \ (\sigma > \beta)$ | | iters | |
|---|---|---|---|---|
| 512 | 6e-08 (19) | 1e-10 (99) | 69 | 725 |
| 1024 | 6e-08 (19) | 1e-10 (99) | 70 | 781 |
| 2048 | 6e-08 (19) | 1e-10 (99) | 68 | 763 |
| 4096 | 6e-08 (19) | 1e-10 (99) | 71 | 713 |

- Fixed $\beta$ : CG mesh-independent
- Fixed mesh : CG $\beta$-dependent
- $\beta$ depends on frequency information that we need to recover
  - Truncation noise $\rightarrow \beta \geq h^2$

# Multigrid

Solve $Hu = g$, and let $u_f^0$ intial guess
$\boldsymbol{S}$ smoothing, $\boldsymbol{R}$ restriction, $\boldsymbol{P}$ prolongation

| | |
|---|---|
| Presmooth Fine | $u_f = \boldsymbol{S}(H_f, u_f^0, g_f)$ |
| Compute Residual | $r_f = g_f - Hu_f$ |
| Restrict | $r_c = \boldsymbol{R} r_f$ |
| Exact coarse | $H_c e_c = r_c$ |
| Correct | $u_f = u_f + \boldsymbol{P} e_c$ |
| Postsmooth Fine | $u_f = \boldsymbol{S}(H_f, u_f, g_f)$ |

# Challenges in designing fast solvers for reduced Hessians

- Typically dense
- Typically only "MatVec" available
- Positivity comes in many flavors
  - I + Compact
  - Compact
  - Differential
- Multigrid available for first and third
  - Little available in the middle category
  - Inverse problems when rhs $\notin$ range(Hessian)

| | |
|---|---|
| **Smoothers** | (coarse/fine grids work partitioning) |
| **Transfer operators** | (do not contaminate spectrum) |
| **Globalization** | (non-convex problems) |

# Related work on multigrid

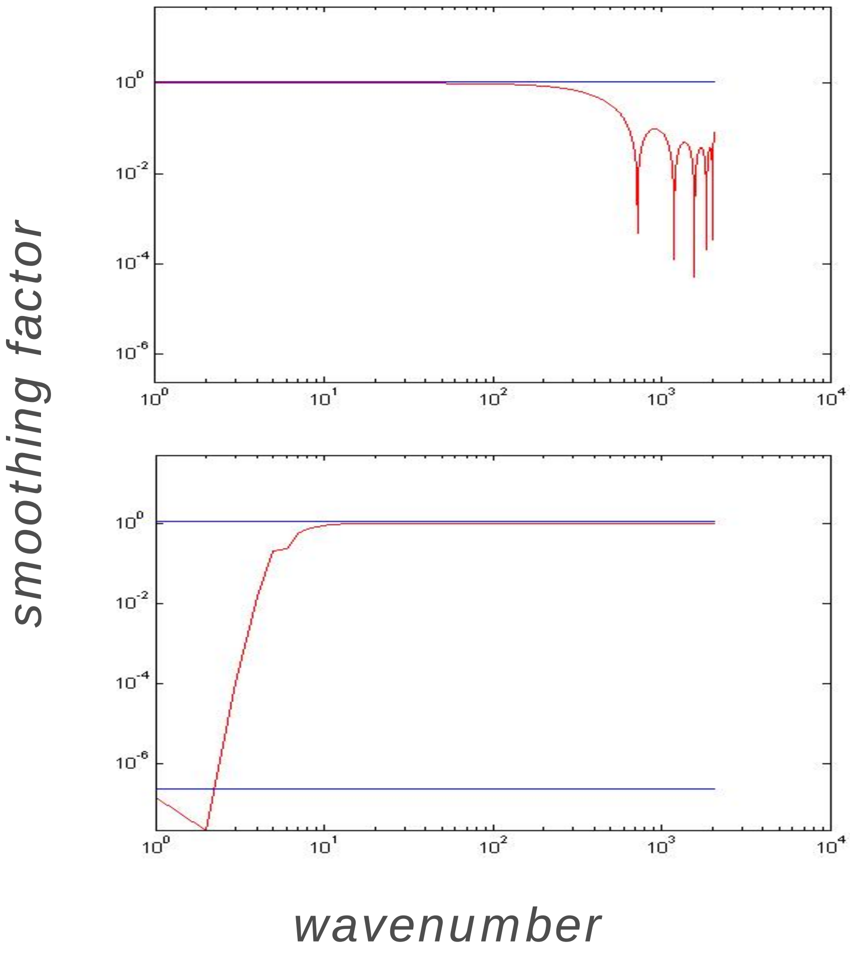
- Multigrid - elliptic PDEs
  - *Brandt, Braess, Bramble, Hackbusch*

- Multigrid – second kind Fredholm
  - *Hackbusch, Hemker & Schippers*

- Multigrid for optimization
  - *Ascher & Haber & Oldenburg, Borzi, Borzi & Kunisch, Borzi & Griesse, Chavent, Dreyer & Maar & Schultz, Draganescu, Hanke & Vogel, Lewis & Nash, Kaltenbacher, King, Kunoth, Ta'asan, Tau & Xu, Vogel, Toint*

- Large-scale parallel multigrid/cascade
  - *Akcelic, Biros & Ghattas, Akcelic* et al.

# CG performance



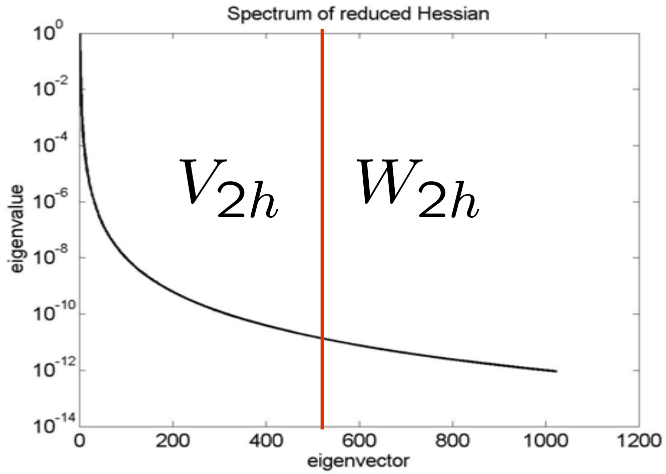

- Laplacian
- Reduced Hessian

# Smoothers for optimization-related operators

- *Hackbush* smoother(s) for 2nd-kind Fredholm
  - Not good in practice, especially for inverse problems
- *Kaltenbacher-King* smoothers
  - Easier to implement but not scalable as regularization $\to 0$
- *Borzi-Griesse* pointwise space-time multigrid
  - Collective, but expensive for general regularization
- Time domain-decomposition
  - *Lions, Hienkenschloss, Maday*

# Design of Smoothers

$$\sigma_k = \mathcal{O}\left(\frac{1}{k^4}\right) \quad \nu_k = \sin(k\pi x).$$




- Orthogonal decomposition (King) of the finite dimensional space

$$V_h = V_{2h} \oplus W_{2h}$$

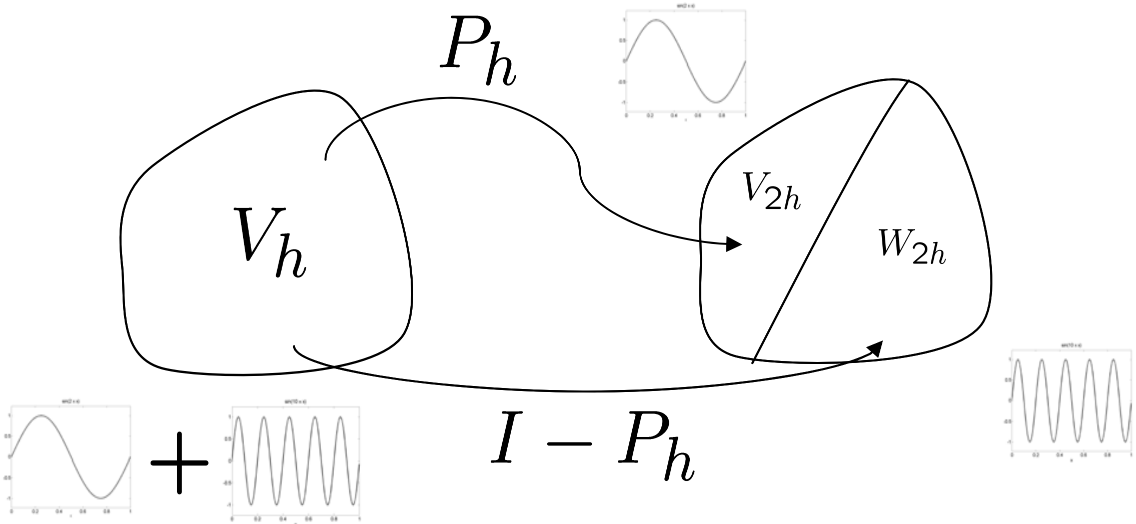
  - relatively low frequency $\longrightarrow$ $V_{2h}$
  - relatively high frequency $\longrightarrow$ $W_{2h}$

- Smoothing iterations on the relatively high-frequency subspace

# Orthogonal decomposition

- Orthogonal decomposition of space

$$u = u_s + u_o \text{ where } u_s \in V_{2h}, u_o \in W_{2h}$$




$$(I - P_h + P_h)H^h(I - P_h + P_h)(u) = g$$

Smoothing equation

$$(I - P_h)H^h u_o = (I - P_h)g$$

Coarse-grid equation

$$P_h H^h u_s = P_h g$$

# Solving smoothing equation $(I - P_h)H^h u_o = (I - P_h)g$

- Smoother
  - Two step stationary iterative scheme by Frankel
  - Constant smoothing factors independent of mesh size and regularization parameter can be derived
- Orthogonal decomposition
  - Discrete sine transforms
  - O(NlogN) operations

Smoothing factor :

$$\mu_k = ((\alpha - \frac{2\alpha\sigma_k}{\lambda_{\min} + \lambda_{\max}})(1 - \frac{2\sigma_k}{\lambda_{\min} + \lambda_{\max}}) + (1 - \alpha)),$$

$$\mu = \max(\mu_k) = 0.29 \text{ for } \frac{\lambda_{\min}}{\lambda_{\max}} = 0.25$$

# Approximate Hessian

Forward

$$\frac{\partial y^n}{\partial t} - \nu \Delta_d y^n = u^n(x)1(t) + \nu \Delta_o y^p[u^p]$$

Adjoint

$$-\frac{\partial p^n}{\partial t} - \nu \Delta_d p^n = -y^n + \nu \Delta_o p^p[u^p]$$

Control

$$\beta u^n - \int_T p^p \, dt = 0$$

# 2 step stationary scheme + Galerkin operator

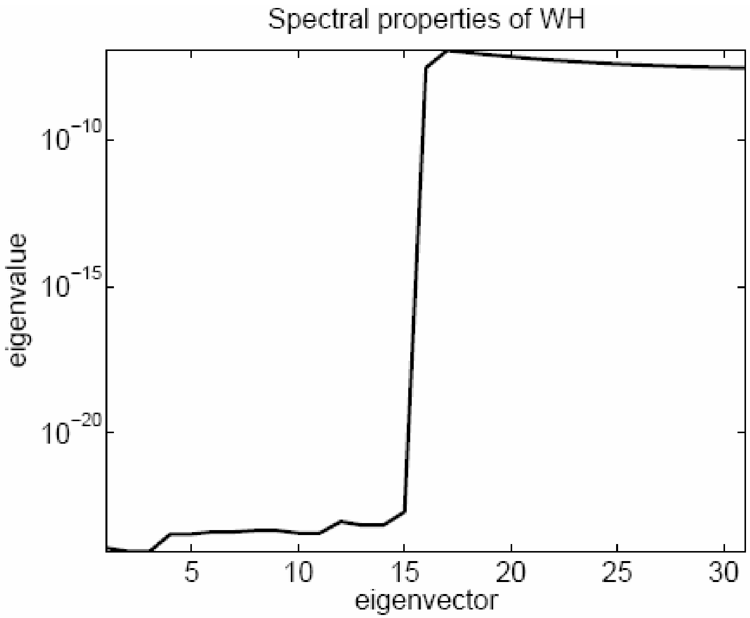
- Experiment : Number of V(2,2) cycles for zero regularization parameter for $\nu$ = 1, 0.01.

| level | $V_1$ | $V_{0.01}$ |
|---|---|---|
| 4 | 16 | 22 |
| 5 | 22 | 27 |
| 6 | 25 | 32 |
| 7 | 26 | 34 |
| 8 | 24 | 33 |

- Mesh-independent and $\beta$-independent but expensive because of Galerkin operator.

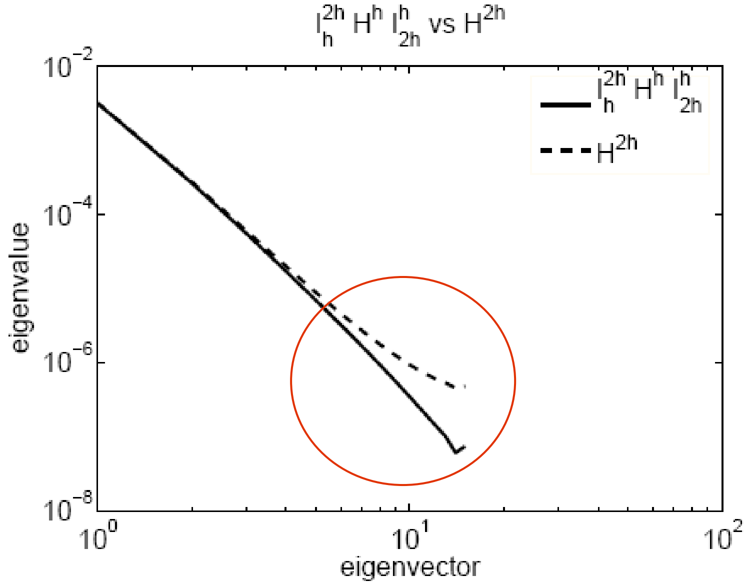
# Smoother + Discretization at coarser grids

Smoother acts on




*Smoother removes high frequency errors*

Coarse grid operator acts on




*Coarse-grid correction does not remove all low frequency errors*

# Non-Galerkin approximation

$\nu = 1.0$

| $N_s$ | $\beta$ | $\mathrm{M}_{\mathrm{sec}}^{-1}$ | $\tilde{\mathrm{M}}_{\mathrm{sec}}^{-1}$ | $\mathrm{M}_{\mathrm{stc}}^{-1}$ | $\tilde{\mathrm{M}}_{\mathrm{stc}}^{-1}$ |
|---|---|---|---|---|---|
| 31 | 5e-07 (31) | 10 | 10 | 10 | 10 |
| 63 | 1e-07 (44) | 13 | 13 | 13 | 17 |
| 127 | 3e-08 (63) | 13 | 14 | 14 | 16 |
| 255 | 7e-09 (89) | 13 | 19 | 13 | 16 |
| 511 | 2e-09 (127) | 15 | 18 | 15 | 17 |
| 1023 | 5e-10 (180) | 15 | 17 | 15 | 17 |

- Number of PCG iterations with V(2,2) cycles as preconditioner are mesh-independent and β-independent

# Non-constant coefficient case

1D parabolic PDE $\quad \Omega = (0,1),\ T = (0,1)$

$$\min \frac{1}{2}\int_T \int_\Omega (y - y^*)^2 + \frac{\beta}{2}\int_\Omega u^2$$

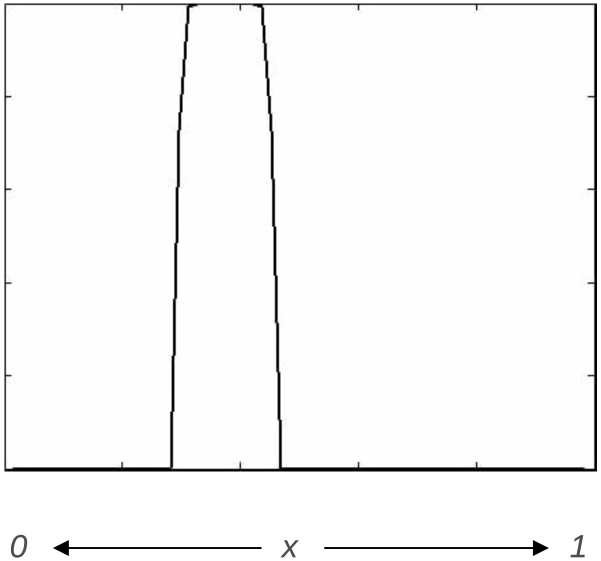
subject to

$$\frac{\partial y}{\partial t} - \nu \Delta y + a(x,t)y + b(x,t)u(x) = 0$$
$$y(x,0) = 0,\ x \in \Omega, \quad y(0,t) = y(1,t) = 0.$$

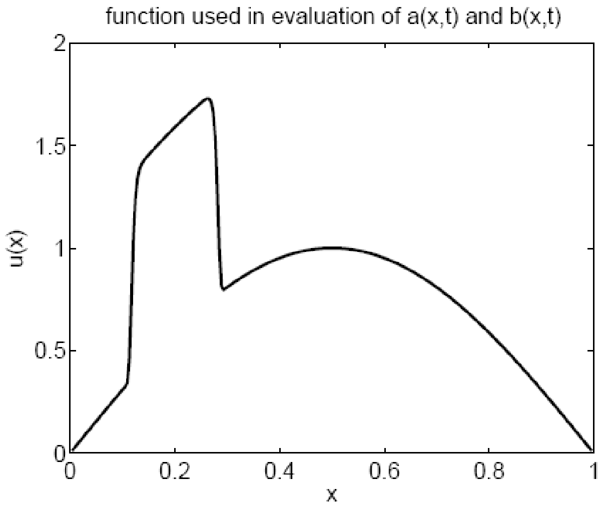
- a(x,t) and b(x,t) are smooth and bounded
- sines are no longer the eigenvectors

# Variable coefficients

$$\widehat{y}(x,t)$$




$$\widehat{u}(x)$$



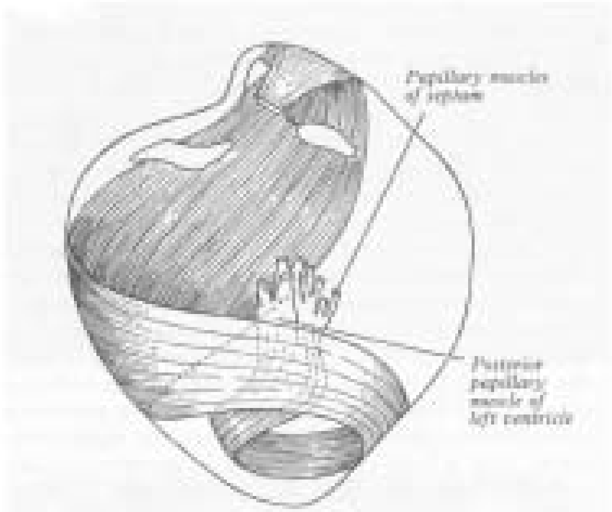

$$a(x,t) = 2\widehat{y}\widehat{u}, b(x,t) = \widehat{y}^2$$

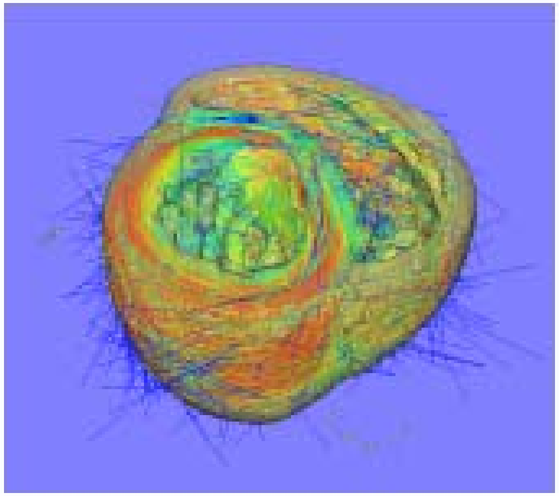
| $N_s$ | $\beta$ | $M_{\text{sec}}^{-1}$ | $M_{\text{stc}}^{-1}$ |
|---|---|---|---|
| 31 | 2e-06 | 13 | 14 |
| 63 | 5e-07 | 14 | 16 |
| 127 | 1e-07 | 16 | 18 |
| 255 | 3e-08 | 19 | 20 |

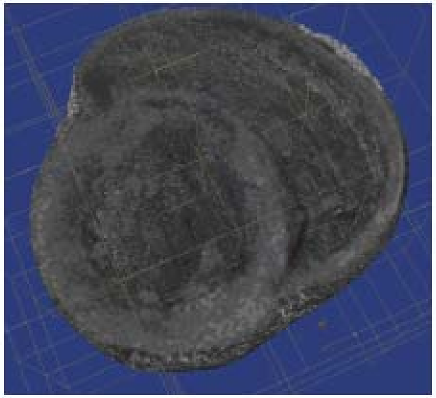
Penn

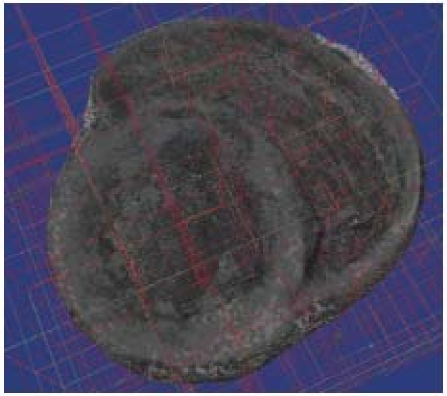
3D

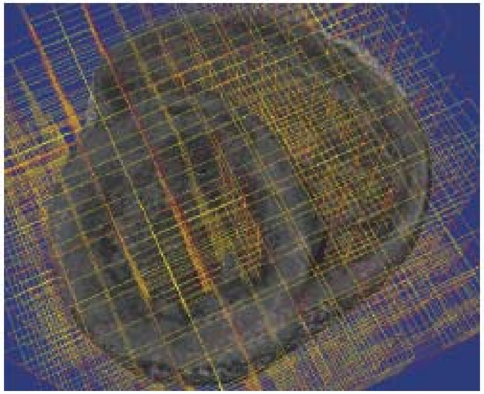
# Electrophysiology

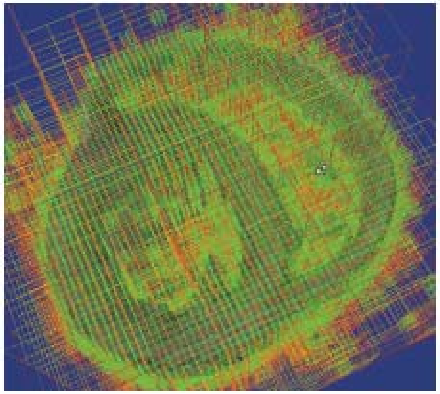










# Octree data-structures

- Construction in parallel
  - Top – down approach
    - Load balancing is an issue
- Balancing in parallel
  - ripple propagation
    - Iterative communication and parallel searches
    - Poor scalability
- Finite elements and handling hanging nodes
  - Projection schemes ($B^TAB$)
    - Need multiple passes to do a matvec
    - More memory requirement to store hanging node information

# Novel approach

- Construction in parallel
  - Bottom-up approach
- Balancing in parallel
  - By *a-priori* communication
    - no iterative communication and parallel searches
- Finite elements and handling hanging nodes
  - Pre-compute element type
    - Single pass to do one matvec
    - Compress both octree information and connectivity information which makes it cache efficient

# Performance

- Matvec
  - For uniform distribution matvec takes almost the same time as regular grid
  - For Gaussian distribution it takes only twice the time as regular grid

*Time to mesh and perform matvec on a single processor*

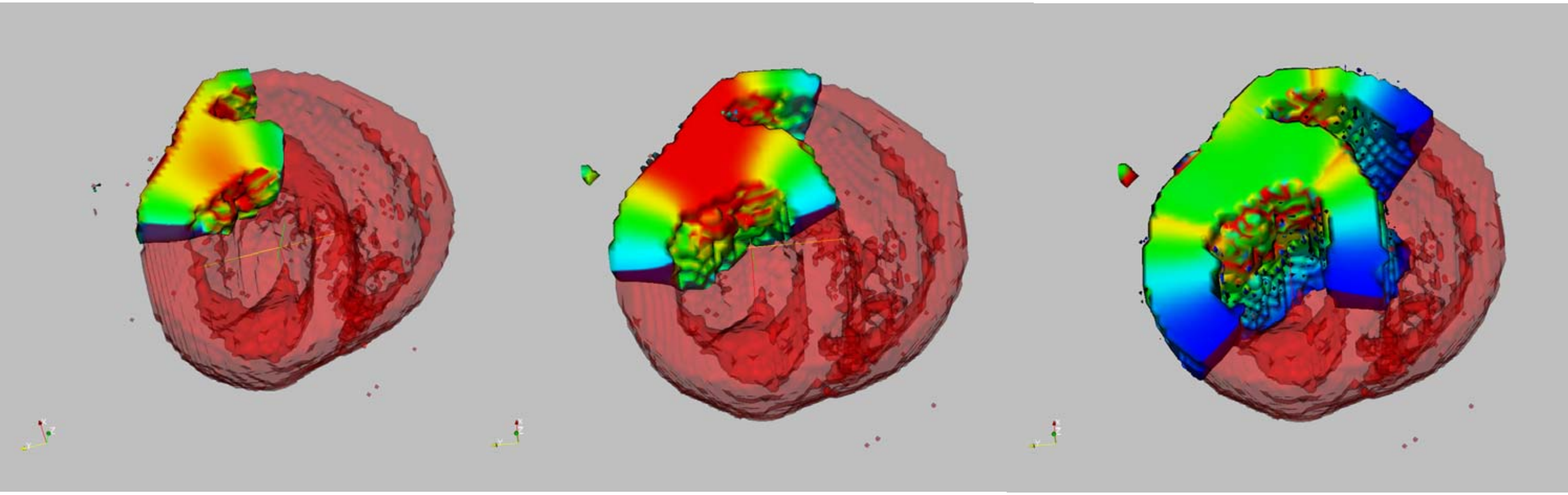
| Problem Size | Regular Grid MatVec | Octree Mesh | | | |
|---|---|---|---|---|---|
| | | Uniform Distribution | | Gaussian Distribution | |
| | | Meshing | MatVec | Meshing | MatVec |
| 256K | 2.1483 | 5.26415 | 2.3994 | 6.13723 | 5.5050 |
| 512K | 4.3557 | 10.3013 | 5.2296 | 12.7178 | 10.560 |
| 1M | 8.1822 | 21.4262 | 9.9921 | 25.8099 | 20.753 |
| 2M | 17.247 | 44.2938 | 19.119 | 53.618 | 43.046 |
| 4M | 34.251 | 90.1725 | 39.541 | 109.351 | 82.198 |

# Performance

- Construction and balancing
  - Parallel time complexity : $\mathcal{O}(n/n_p \log(n/n_p) + n_p \log n_p)$
  - Storage complexity : $\mathcal{O}(n)$


| | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 | 256 | 512 | 1024 | 2048 | 4096 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Octree Construction | 1.0 | 1.5 | 1.4 | 1.4 | 1.4 | 1.7 | 1.6 | 1.7 | 1.8 | 1.8 | 2.4 | 3.8 | 6.75 |
| Octree Balancing | 5.3 | 8.4 | 8.8 | 10.7 | 12.0 | 12.6 | 13.6 | 13.6 | 14.2 | 15.6 | 18.0 | 25.2 | 39.48 |
| Mesh Construction | 20.9 | 26.9 | 25.0 | 27.8 | 29.5 | 31.3 | 29.2 | 31.3 | 33.8 | 33.9 | 34.9 | 37.1 | 41.12 |
| MatVec (5) | 27.3 | 31.1 | 32.9 | 32.5 | 39.7 | 41.3 | 39.5 | 41.4 | 51.1 | 48.4 | 45.5 | 49.2 | 48.49 |

# 3D forward solver



# Summary

- Computationally demanding forward problems
  - Billions of states, millions of opt parameters
- Need for fast Hessian solvers
- Special structure of Hessian for inverse problems
- Multigrid
  - Proposed new smoother
- New octree data-structures
- Ongoing data structures
  - Scalable 3D implementation of the inversion algorithm

- url
  - www.seas.upenn.edu/~biros/papers/
- Octrees
  - OctreeBalance21.pdf
  - octreeFem.pdf
- Inversion
  - h1dinv.pdf
- Integral equation solvers
  - heat1d.pdf
  - heat2d.pdf