# The chaotic nature of faster gradient descent methods

**Uri Ascher**

Department of Computer Science
University of British Columbia

ascher@cs.ubc.ca
www.cs.ubc.ca/∼ascher

with **Kees van den Doel**, Hui Huang, Benar Svaiter

# Outline

- Motivation
  - Box constrained convex quadratic optimization
  - Unconstrained gradient descent
  - Artificial time
  - Faster gradient descent examples
  - Finite time regularization

# Box constrained convex quadratic optimization

[Friedlander & van den Berg, '08; Figueiredo, Nowak & Wright, '07; Dai & Fletcher '05]:

To find sparse solutions for **compressed sensing** etc. consider

$$\min_{\mathbf{x}} \|\mathcal{A}\mathbf{x} - \mathbf{y}\|^2, \quad s.t. \ \|\mathbf{x}\|_1 \leq tol,$$

i.e. minimize convex quadratic subject to box constraints, and use a **fast gradient descent** method.

# Box constrained convex quadratic optimization

[Friedlander & van den Berg, '08; Figueiredo, Nowak & Wright, '07; Dai & Fletcher '05]:

To find sparse solutions for compressed sensing etc. consider

$$\min_{\mathbf{x}} \|\mathcal{A}\mathbf{x} - \mathbf{y}\|^2, \quad s.t. \ \|\mathbf{x}\|_1 \leq tol,$$

i.e. minimize convex quadratic subject to box constraints, and use a **fast gradient descent** method.

**??** But is there a fast gradient descent method **??**

# Unconstrained gradient descent

For the basic problem

$$A\mathbf{x} = \mathbf{b},$$

$A$ symmetric positive definite,

well known that steepest descent (SD) is very slow, conjugate gradient (CG) much faster.

**Gradient descent**

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{r}_k, \quad \mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k.$$

# Steepest and lagged steepest descent

**Gradient descent** $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{r}_k, \quad \mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k.$

**Steepest descent**

$$\alpha_k^{SD} = \frac{\mathbf{r}_k^T \mathbf{r}_k}{\mathbf{r}_k^T A \mathbf{r}_k} \equiv \frac{(\mathbf{r}_k, \mathbf{r}_k)}{(\mathbf{r}_k, A\mathbf{r}_k)}.$$

Claim: much faster to use non-monotonically convergent **Lagged steepest descent** (LSD) [Barzilai & Borwein, '88]

$$\alpha_k^{LSD} = \frac{(\mathbf{r}_{k-1}, \mathbf{r}_{k-1})}{(\mathbf{r}_{k-1}, A\mathbf{r}_{k-1})}.$$

# Minimization and artificial time

Unconstrained minimization:

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad \text{necessary } \boldsymbol{\nabla} f(\mathbf{x}) = \mathbf{0}.$$

# Minimization and artificial time

Unconstrained minimization:

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad \text{necessary } \boldsymbol{\nabla} f(\mathbf{x}) = \mathbf{0}.$$

Convex quadratic case:

$$f(\mathbf{x}) = \frac{1}{2}(\mathbf{x}, A\mathbf{x}) - (\mathbf{b}, \mathbf{x}), \quad \boldsymbol{\nabla} f(\mathbf{x}) = A\mathbf{x} - \mathbf{b} = -\mathbf{r}.$$

Steepest descent: gradient descent with exact line search wrto $f$.

# Minimization and artificial time

Unconstrained minimization:

$$\min_{\mathbf{x}} f(\mathbf{x}), \quad \text{necessary } \boldsymbol{\nabla} f(\mathbf{x}) = \mathbf{0}.$$

Convex quadratic case:

$$f(\mathbf{x}) = \frac{1}{2}(\mathbf{x}, A\mathbf{x}) - (\mathbf{b}, \mathbf{x}), \quad \boldsymbol{\nabla} f(\mathbf{x}) = A\mathbf{x} - \mathbf{b} = -\mathbf{r}.$$

Steepest descent: gradient descent with exact line search wrto $f$.

Necessary condition may be considered as steady state for ODE:

$$\frac{d\mathbf{x}}{dt} = -\boldsymbol{\nabla} f(\mathbf{x}).$$

# Artificial time DE

$$\frac{d\mathbf{x}}{dt} = -\boldsymbol{\nabla} f(\mathbf{x}).$$

**Forward Euler** discretization

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \alpha_k \boldsymbol{\nabla} f(\mathbf{x}_k).$$

Equivalent to gradient descent.

How to choose step size? What about absolute stability?

# Faster gradient descent

$$u_t = \Delta u + 1 \quad 0 < x, y < 1, \; t \geq 0,$$

+ homogeneous Dirichlet BC.

Discretize in space on a uniform mesh and call unknowns $\mathbf{x} \in \mathbb{R}^m$. Apply forward Euler in time, stopping when $\|\mathbf{r}_k\| \leq 10^{-6}\|\mathbf{r}_0\|$.

| $m$ | SD | CG |
|---|---|---|
| 49 | 167 | 9 |
| 225 | 702 | 24 |
| 961 | 2,859 | 50 |
| 3969 | 11,517 | 100 |

Step (iteration) counts for the heat $\Rightarrow$ Poisson equation

# Faster gradient descent

$$u_t = \Delta u + 1 \quad 0 < x, y < 1, \; t \geq 0,$$

+ homogeneous BC.

Discretize in space on a uniform mesh and call unknowns $\mathbf{x} \in \mathbb{R}^m$. Apply forward Euler in time, stopping when $\|\mathbf{r}_k\| \leq 10^{-6}\|\mathbf{r}_0\|$.

| $m$ | SD | LSD | CG |
|---|---|---|---|
| 49 | 167 | 40 | 9 |
| 225 | 702 | 72 | 24 |
| 961 | 2,859 | 240 | 50 |
| 3969 | 11,517 | 356 | 100 |

Step (iteration) counts for the heat $\Rightarrow$ Poisson equation

# LSD vs CG

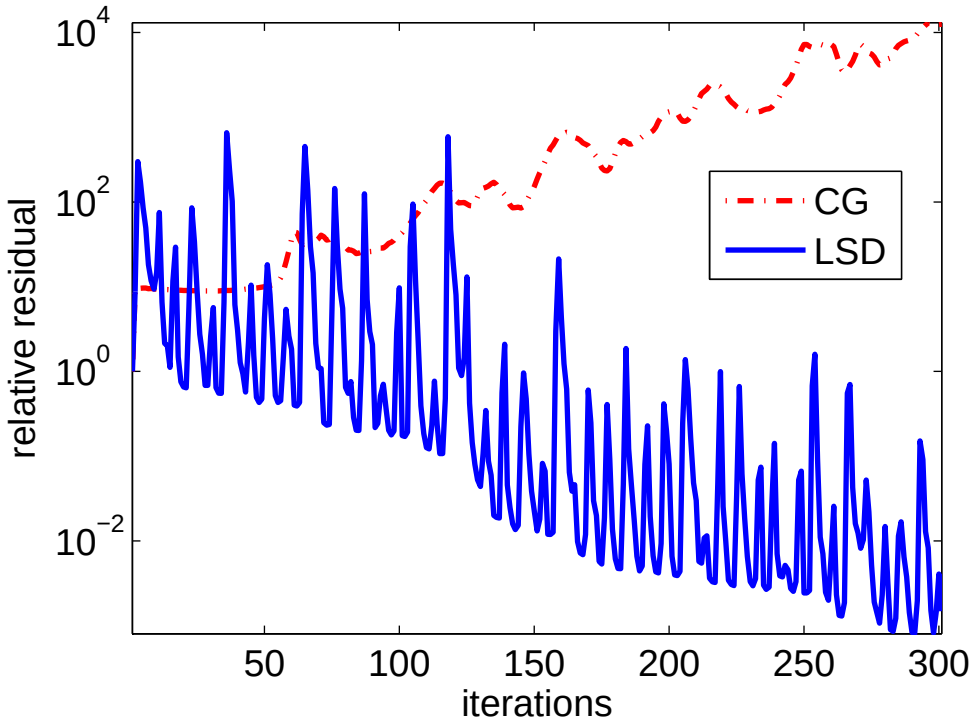
*CG clearly majorizes any gradient descent method, including LSD*

# LSD vs CG

*CG clearly majorizes any gradient descent method, including LSD*

**unless** matrix-vector products are performed inaccurately!

**Example**: replace the product $A\mathbf{v}$ for any given vector $\mathbf{v}$ by $A(\mathbf{v} + 0.005/(1 + \mathbf{v}^2))$.




CG and LSD applied to a small perturbation of the model Poisson problem, $m = 961$.

# Smoothing and finite time regularization

Consider linear problem $J\mathbf{x} = \mathbf{y}$, $J$ constant and ill-conditioned.

Artificial time ODE

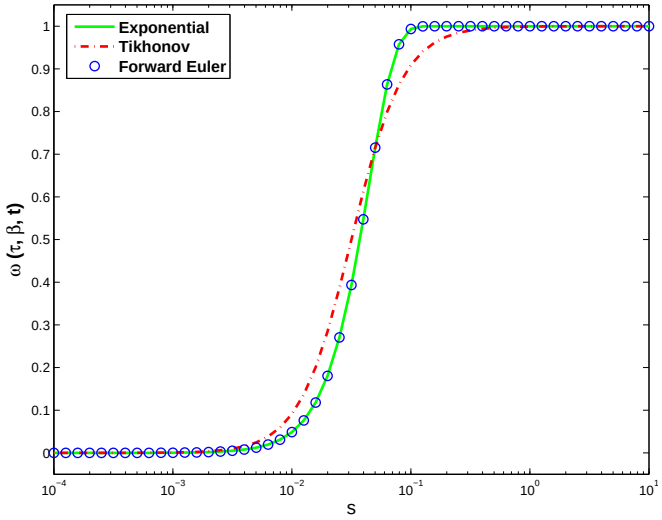
$$\frac{d\mathbf{x}}{dt} = -J^T(J\mathbf{x} - \mathbf{y}) \equiv \mathbf{b} - A\mathbf{x}.$$

Using SVD, regularization replaces singular values $s_i^{-1}$ by $\omega(s_i^2)s_i^{-1}$, where

$$\omega(s) = 1 - e^{-ts}.$$

So, the effect of small singular values gets dampened while large ones remain almost intact for appropriate, **finite time**.

# Filter functions




Exponential filter $\omega(s) = 1 - e^{-ts}$ and Tikhonov filter $\omega(s) = \frac{s}{s+\beta}$ for $t\beta = 1/2$.

**Question**: Does this regularization effect still hold when ODE is approximately integrated with large steps? Is there an advantage in doing so?

# Outline

- Motivation
- Linear positive definite systems
- Chaos
- Regularization effects of gradient descent / artificial time
- Conclusions

# Outline

- Linear positive definite systems
  - Step size choices for gradient descent / forward Euler
  - Slowness of greedy algorithms
  - Faster step size selection

$$A\mathbf{x} = \mathbf{b}$$

# Step size choices

$$\alpha_k^{SD} = \frac{\|\mathbf{r}_k\|^2}{\|\mathbf{r}_k\|_A^2}$$

Another **greedy algorithm**: exact line search wrto $\|\mathbf{r}\|^2$

$$\alpha_k^{OM} = \frac{(\mathbf{r}_k, A\mathbf{r}_k)}{(A\mathbf{r}_k, A\mathbf{r}_k)}.$$

# Orthomin (OM) and SD combinations

$$\alpha_k^{SD} = \frac{\|\mathbf{r}_k\|^2}{\|\mathbf{r}_k\|_A^2},$$

$$\alpha_k^{OM} = \frac{\|\mathbf{r}_k\|_A^2}{\|A\mathbf{r}_k\|^2} = \frac{\|\mathbf{r}_k\|_A^2}{\|\mathbf{r}_k\|_{A^2}^2}.$$

Combinations:

$$\alpha_k^{SD/OM} = \begin{cases} \alpha_k^{SD} & \text{k even} \\ \alpha_k^{OM} & \text{k odd} \end{cases}.$$

# Half lagged and relaxed steepest descent

- **Half lagged (HLSD)**

  Like LSD, a two-step method: use same $\alpha_{2j}^{SD}$ for next step: $\alpha_{2j+1}^{HLSD} = \alpha_{2j}^{HLSD}$.

  [Friedlander, Martinez, Molina & Raydan, '99; Raydan & Svaiter, '02]

- **$\omega$-relaxed (SD($\omega$))**

  $$\alpha_k = \omega \alpha_k^{SD}$$

  for a fixed parameter $0 < \omega < 1$.

# Numerical example: heat $\Rightarrow$ Poisson equation

Iteration counts for more step-size selections, same model problem

| $m$ | SD | LSD |
|---|---|---|
| 49 | 167 | 40 |
| 225 | 702 | 72 |
| 961 | 2,859 | 240 |
| 3969 | 11,517 | 356 |

# Numerical example: heat $\Rightarrow$ Poisson equation

Iteration counts for more step-size selections, same model problem

| $m$ | SD | OM | LSD |
|---|---|---|---|
| 49 | 167 | 169 | 40 |
| 225 | 702 | 696 | 72 |
| 961 | 2,859 | 2,811 | 240 |
| 3969 | 11,517 | 11,279 | 356 |

# Numerical example: heat $\Rightarrow$ Poisson equation

Iteration counts for more step-size selections, same model problem

| $m$ | SD | OM | SD/OM | LSD |
|---|---|---|---|---|
| 49 | 167 | 169 | 46 | 40 |
| 225 | 702 | 696 | 88 | 72 |
| 961 | 2,859 | 2,811 | 276 | 240 |
| 3969 | 11,517 | 11,279 | 878 | 356 |

# Numerical example: heat $\Rightarrow$ Poisson equation

Iteration counts for more step-size selections, same model problem

| $m$ | SD | OM | SD/OM | SD(0.9) | LSD |
|---|---|---|---|---|---|
| 49 | 167 | 169 | 46 | 52 | 40 |
| 225 | 702 | 696 | 88 | 136 | 72 |
| 961 | 2,859 | 2,811 | 276 | 290 | 240 |
| 3969 | 11,517 | 11,279 | 878 | 824 | 356 |

# Numerical example: heat $\Rightarrow$ Poisson equation

Iteration counts for more step-size selections, same model problem

| $m$ | SD | OM | SD/OM | SD(0.9) | LSD | HLSD |
|---|---|---|---|---|---|---|
| 49 | 167 | 169 | 46 | 52 | 40 | 59 |
| 225 | 702 | 696 | 88 | 136 | 72 | 67 |
| 961 | 2,859 | 2,811 | 276 | 290 | 240 | 142 |
| 3969 | 11,517 | 11,279 | 878 | 824 | 356 | 590 |

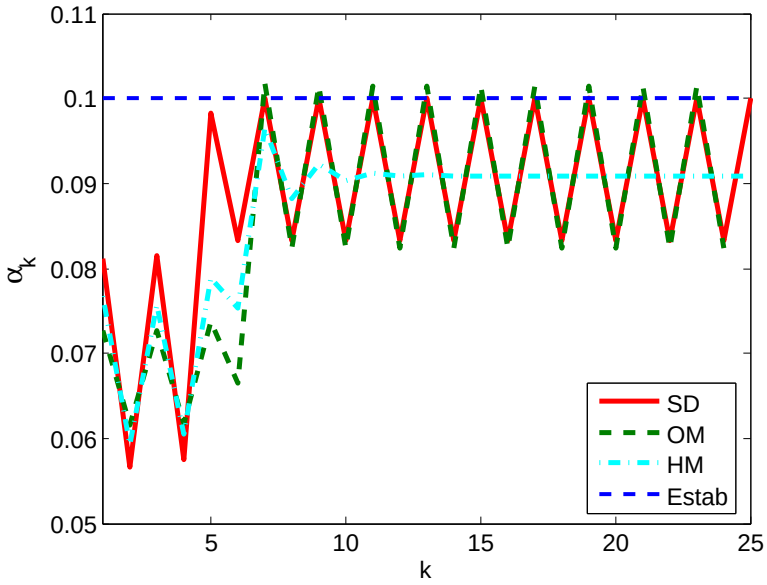
# It's not true that:

- A fast method must be multi-step
- A fast method must be non-monotonic
- LSD works because it's a secant approximation

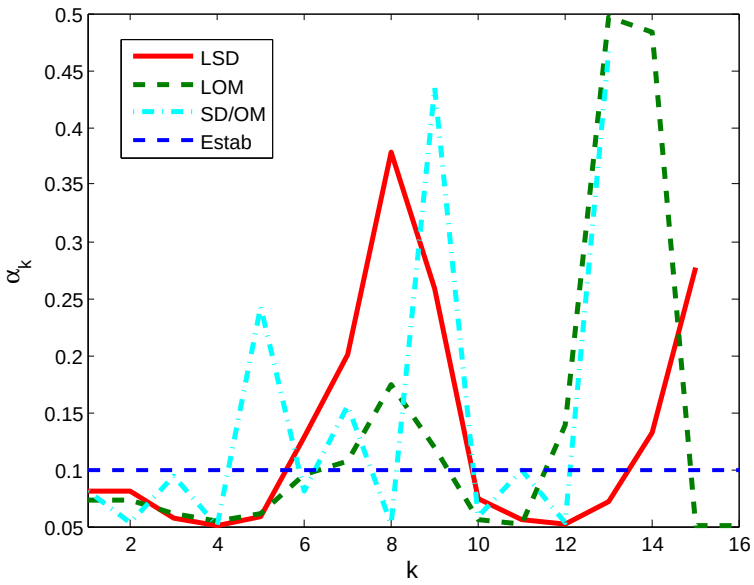
| $m$ | SD | OM | SD/OM | SD(0.9) | LSD | HLSD |
|---|---|---|---|---|---|---|
| 49 | 167 | 169 | 46 | 52 | 40 | 59 |
| 225 | 702 | 696 | 88 | 136 | 72 | 67 |
| 961 | 2,859 | 2,811 | 276 | 290 | 240 | 142 |
| 3969 | 11,517 | 11,279 | 878 | 824 | 356 | 590 |

# Step size dynamics

Step size sequences for $A = \mathrm{diag}(20, 10, 2, 1)$



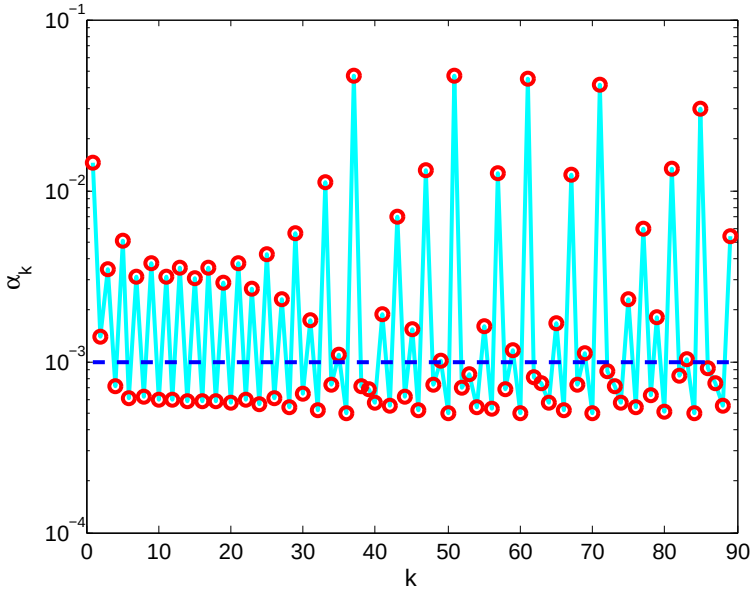

(a) Greedy step sizes



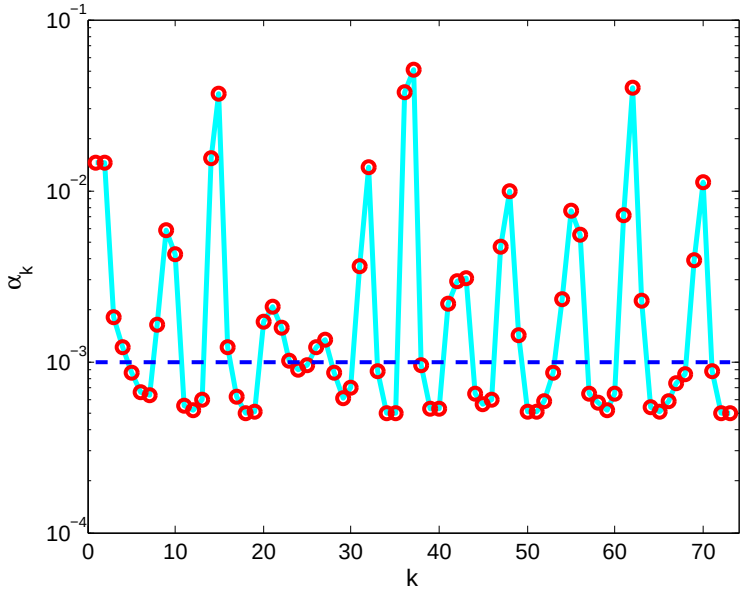

(b) Wilder step sizes

# Step size dynamics

Step size sequences for heat $\Rightarrow$ Poisson eqn example, $m = 225$. Note large violation of forward Euler stability limit.




(c) Step sizes SD/OR




(d) Step sizes LSD

# Slowness of the greedy algorithms

Denote eigenvalues of $A$ by $\lambda_1 > \lambda_2 > \cdots > \lambda_m > 0$.

## Case of constant (uniform) step size

**Theorem** Barring special initial conditions, the fastest reduction in residual norm towards steady state using forward Euler with a constant step size is obtained using the step size

$$\alpha^* = \frac{2}{\lambda_1 + \lambda_m}.$$

For this step size the number of steps required to reduce the residual error by a constant amount is proportional to the condition number $\kappa = cond(A) = \frac{\lambda_1}{\lambda_m}$.

# Residuals' contraction

Indeed, write iteration in terms of residual:

$$\mathbf{r}_{k+1} = (I - \alpha_k A)\mathbf{r}_k.$$

Assume wlog a diagonal $A$. Then

$$r_i^{(k+1)} = (1 - \alpha_k \lambda_i) r_i^{(k)}.$$

So

$$r_m^{(k+1)} = (1 - \alpha^* \lambda_m) r_m^{(k)} = \left( \frac{\lambda_1 - \lambda_m}{\lambda_1 + \lambda_m} \right) r_m^{(k)} = \left( \frac{\kappa - 1}{\kappa + 1} \right) r_m^{(k)}.$$

# Slowness of the greedy algorithms cont.

## Case of variable step size

**Theorem** Barring special initial conditions:

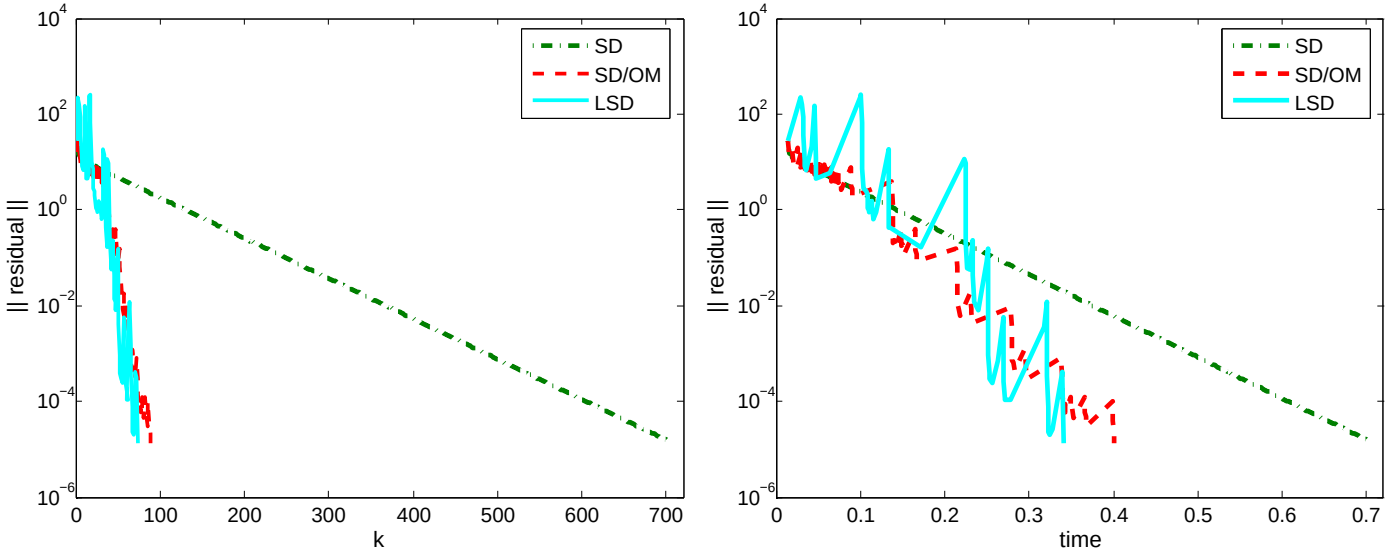
1. For SD there are constants satisfying $\frac{2}{\beta_0^{-1}+\beta_1^{-1}} = \alpha^*$ such that as $j \to \infty$, $\alpha_{2j} \to \beta_0$ ; $\alpha_{2j+1} \to \beta_1$.

2. Similarly for OM.

[Akaike, '59]

# Faster step size selection

Both one-step and multi-step strategies can yield faster convergence.
Payment: **non-monotonic convergence**.




(e) $||\mathbf{r}||$ as a function of step #

(f) $||\mathbf{r}||$ as a function of time

Residual norms and step size sequences.

# Outline

- Motivation
- Linear positive definite systems
- Chaos
- Regularization effects of gradient descent / artificial time in applications
- Conclusions

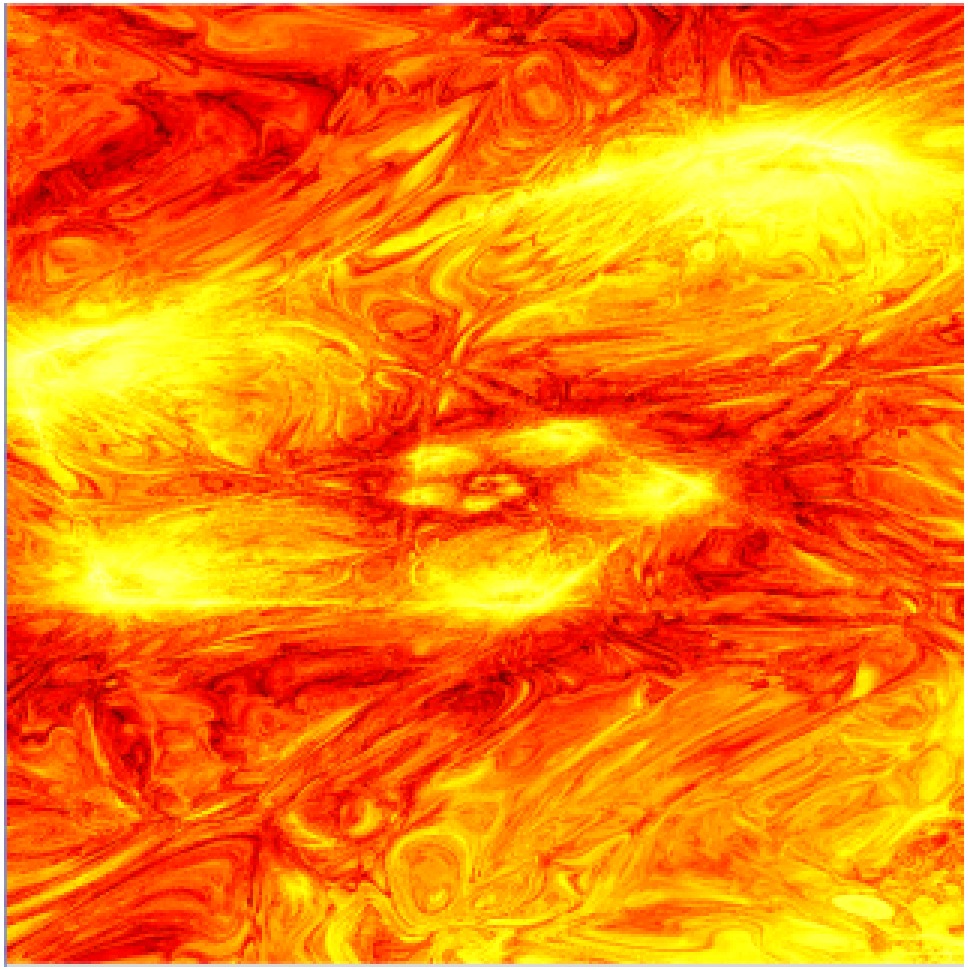
# Outline

- Chaos
  - Sensitivity to initial guess
  - Lyapunov exponents
  - SD($\omega$): perils of greed and chaos around the corner
  - LSD and HLSD

# Sensitivity to initial guess

| $m$ | $\mathbf{x}_0$ | CG | SD | LSD | LSD(2) | HLSD | SD(0.8) |
|---|---|---|---|---|---|---|---|
| 49 | (a) | 10 | 341 | 71 | 69 | 87 | 113 |
| | (b) | 10 | 341 | 77 | 62 | 85 | 120 |
| 225 | (a) | 33 | 1414 | 141 | 179 | 152 | 279 |
| | (b) | 32 | 1414 | 215 | 151 | 143 | 290 |
| 961 | (a) | 71 | 5721 | 412 | 279 | 313 | 585 |
| | (b) | 70 | 5721 | 441 | 417 | 377 | 535 |
| 3969 | (a) | 143 | 22979 | 797 | 712 | 732 | 1331 |
| | (b) | 143 | 22979 | 976 | 567 | 828 | 1459 |

Iteration counts for the model Poisson problem using gradient descent with different step size choices for initial vectors (a) $\mathbf{x}_0 = \mathbf{0}$ and (b) $\mathbf{x}_0 = 10^{-3} \cdot \mathbf{1}$.



Iteration counts for LSD with a perturbation in a plane of extent $10^{-3}$. Brighter color corresponds to higher iteration number.

# Lyapunov exponents

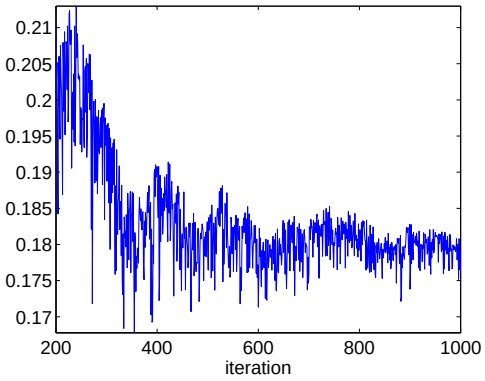
May assume $A = diag(\lambda_1, \ldots, \lambda_m)$.

Define Akaike probability function

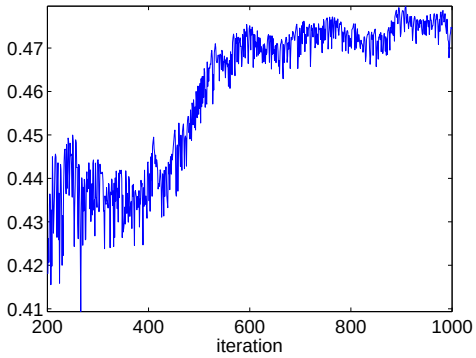
$$p_i = r_i^2 / \|\mathbf{r}\|^2, \quad i = 1, \ldots, m.$$

Consider Jacobian $J_k$ of transformation $\mathbf{p}_{k+1} \leftarrow \mathbf{p}_k$, then define Lyapunov exponent as limit of $\mu_K$, where

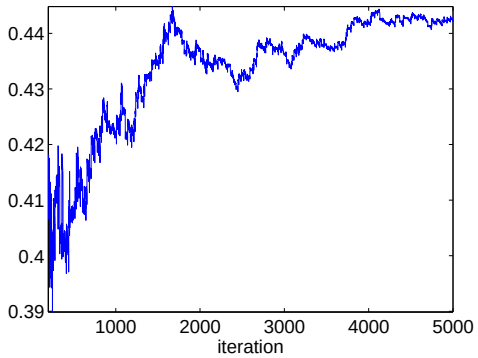
$$J^{(K)} = J_{K-1} \cdots J_1 J_0, \quad \mu_K = \frac{1}{K} \log \left[ \rho(J^{(K)}) \right].$$



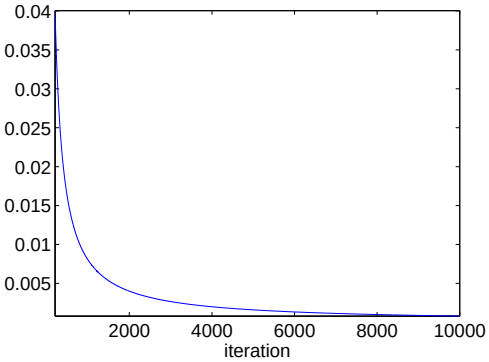

(g) LSD




(h) HLSD




(i) SD(0.8)




(j) SD

Lyapunov exponents $\mu$ for various methods applied to the model Poisson problem with and $m = 49$.

# SD($\omega$)

- $\omega = 1$: step sizes tend to two-cycle; stable; slow.
- $\omega = 0.9$: chaotic dynamical system (even though "close" to SD, converges Q-linearly, clearly one-step, and "dampens" SD); much faster.

Recall for diagonal $A = diag(\lambda_i)$

$$\gamma_k^{SD} = 1/\alpha_k^{SD} = \frac{\sum_i \lambda_i r_i^2}{\sum_i r_i^2} = \sum_i \lambda_i p_i, \quad \alpha_k = \omega \alpha_k^{SD}$$

$$r_i^{(k+1)} = (1 - \alpha_k \lambda_i) r_i^{(k)}, \quad i = 1, \ldots, m.$$

- If $p_i$ are roughly equal then $\lambda_1 p_1$ (or the first few) dominate, so $\alpha_k^{SD} \sim 1/\lambda_1$.

- The corresponding step is effectively a **smoother**: reduces high frequency (large-eigenvalue) residuals much more than low-frequency ones. This effect may be repeated over a few steps.

- The **usual** case: the high frequency residuals become so small that other frequencies factor in, so a much larger step size $\alpha_k^{SD}$ is obtained. The ensuing step is unstable, increasing high frequency probabilities $p_i$. This closes a chaotic cycle.

- The **unusual** case: with SD the high frequency $\lambda_1 p_1$ always dominates. In the limit $p_1$ and $p_m$ alternate (maintaining roughly same order of magnitude) and $p_i \approx 0$ for the rest. Thus chaos (and good news) are avoided.

# LSD and HLSD

Gradient descent iteration:

$$\mathbf{r}_{k-1} = c(A - \gamma_{k-1} I)^{-1} \mathbf{r}_k.$$

Consider as an inverse power iteration for $A$ using the shift $\gamma_{k-1}$.

Therefore, $\mathbf{r}_{k-1}$ better than $\mathbf{r}_k$ approximating the eigenvector associated with the eigenvalue $\lambda_j$ closest to $\gamma_{k-1}$.

**LSD** not only chaotic, also $\gamma_k$ is closer than $\gamma_{k-1}$ to the eigenvalue. "Hence" more effective than one-step methods such as relaxed SD.

# Outline

- Motivation
- Linear positive definite systems
- Chaos
- Regularization effects of gradient descent / artificial time
- Conclusions

# Outline

- Regularization effects of gradient descent / artificial time in applications
  - Image denoising and deblurring in 2D
    **Hui Huang**
  - Shape optimization in 3D for EIT and DC resistivity
    **Kees van den Doel**

# Image denoising in 2D

Find $w(x, y)$ that cleans given image $b(x, y)$.

(k) True image

(l) Given b = with 20% noise

Cameraman $256 \times 256$ with noise added.

# Image deblurring in 2D

Find $w(x, y)$ that sharpens given image $b(x, y)$.

(m) True image

(n) MOTION blur

Boat $256 \times 256$ blurred.

# Huber function regularization

Tikhonov: $\min \ \frac{1}{2}\|Jw - b\|^2 + \beta R(w)$ .

$$R(w) = \int_\Omega \rho(|\boldsymbol{\nabla} w|)$$

Use the Huber function

$$\rho(s) \quad = \quad \begin{cases} s, & s \geq \gamma, \\ s^2/(2\gamma) + \gamma/2, & s < \gamma \end{cases} \quad \Rightarrow$$

$$R_w(w) \quad \leftarrow \quad -\nabla \cdot \left( \min\{\frac{1}{\gamma}, \frac{1}{|\boldsymbol{\nabla} w|}\} \, \boldsymbol{\nabla} w \right)$$

# Selecting the switching parameter

Choose $\gamma$ wisely:

1. Letting $\gamma \to \infty$ obtain least squares (LS)

2. Determine adaptively, close to **total variation** (TV)

$$\gamma = \frac{h}{|\Omega|} \int_\Omega |\boldsymbol{\nabla} w|.$$

**"Our Huber"** [Ascher, Haber & Huang, '06]

# Gradient descent method

Necessary condition for Tikhonov minimum

$$G(w) \equiv J^T(Jw - b) + \beta R_w = \mathbf{0}.$$

# Gradient descent method

Necessary condition for Tikhonov minimum

$$G(w) \equiv J^T(Jw - b) + \beta R_w = \mathbf{0}.$$

Iteration

$$w_{k+1} = w_k - \alpha_k \left( J^T(Jw_k - b) + \beta \hat{R}(w_k) w_k \right), \quad k = 0, 1, 2, \ldots,$$

$$R_w(w) = \hat{R}(w) \cdot w.$$

# Step size selection

Compare the two step size choices obtained by freezing nonlinearities:

$$\text{SD:} \qquad \alpha_k = \frac{(G(w_k))^T G(w_k)}{(G(w_k))^T (J^T J + \beta \hat{R}(w_k)) G(w_k)} \ ,$$

$$\text{LSD:} \qquad \alpha_k = \frac{(G(w_{k-1}))^T G(w_{k-1})}{(G(w_{k-1}))^T (J^T J + \beta \hat{R}(w_{k-1})) G(w_{k-1})} \ .$$

# Image deblurring in 2D

(o) MOTION blur

(p) 33 LSD or 113 SD iters

TYPE = 'MOTION', LEN = 15, THETA = 30, $\eta = 1$, $\beta = 10^{-4}$.

# Image deblurring in 2D

Roughly 1.3 sec per iter for $256 \times 256$ images.

(q) True image

(r) DISK blur

(s) 99 SD or 26 LSD iters

TYPE = 'DISK', RADIUS = 5, $\eta = 1$, $\beta = 10^{-4}$.

## Other related denoising and deblurring methods

- For denoising can use rough tolerance results to switch to another method.

- For deblurring in presence of much noise can try a splitting approach.

In both cases not too many SD iterations are required, and LSD advantage is limited.

## Shape optimization in 3D for EIT and DC resistivity

**Forward problem**:

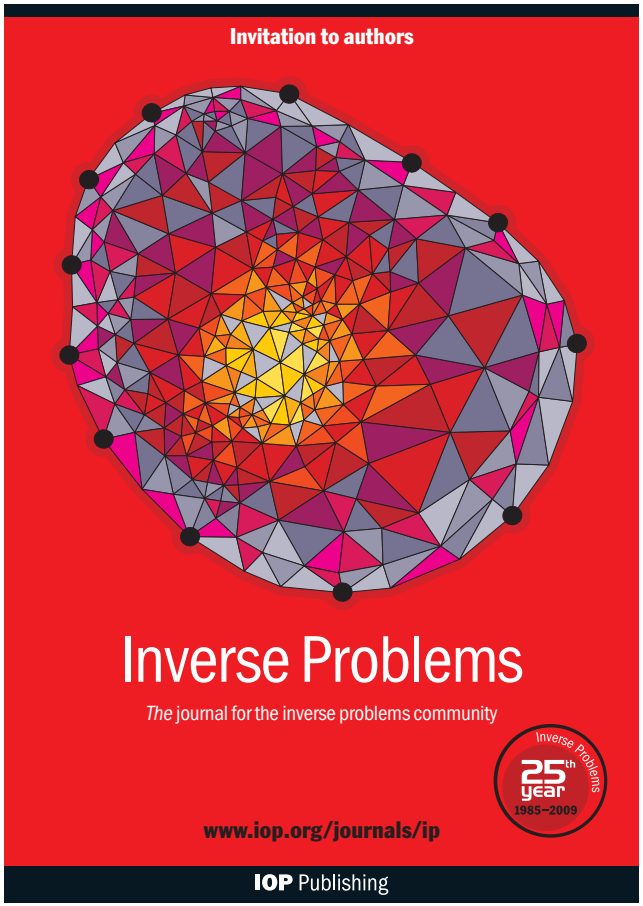
$$\begin{aligned} \nabla \cdot (\sigma \boldsymbol{\nabla} u_j) &= q_j, \quad j = 1, \ldots, s, \\ \frac{\partial u_j}{\partial \nu}|_{\partial\Omega} &= 0. \end{aligned}$$

Domain $\Omega = [0,1]^3$.

Predict data by measuring field at specified locations:
$F(w) = (F_1, \ldots, F_s)^T$, $w = \log \sigma$.

# A 2D picture

[van den Doel, Ascher & Pai, ’08]



# Shape optimization

[van den Doel & Ascher, '07]

Assume $w(x, y, z)$ can take only one of two values at each $(x, y, z)$.

**Inverse problem**: Recover $w$ from measured data $b$.

# Shape optimization

[van den Doel & Ascher, '07]

Assume $w(x, y, z)$ can take only one of two values at each $(x, y, z)$.

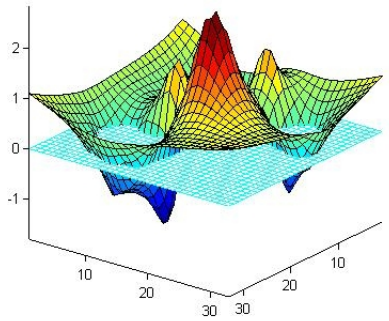
**Inverse problem**: Recover $w$ from measured data $b$.

Thus, **shape optimization**.

# A level set approach

$$
\begin{aligned}
w &= \chi(\psi), \quad e.g. \\
\chi(s) &= \frac{w_I - w_{II}}{2} \tanh(s/h) + \frac{w_I + w_{II}}{2}.
\end{aligned}
$$

Thus, sharpening happens at each iteration $l$ across the $0$- level set of $\psi_l(x, y, z)$.



# Dynamic regularization

Damped **Gauss-Newton** for

$$\min_{\psi} \frac{1}{2} \sum_{j=1}^{s} \|F_j(w(\psi)) - b_j\|^2$$

yields iteration

$$\left( \sum_{j=1}^{s} \hat{J}_j^T \hat{J}_j \right) \delta\psi = -\tau \sum_{j=1}^{s} \hat{J}_j^T (F_j(w(\psi)) - b_j)$$

where

$$\hat{J} = \frac{\partial F}{\partial \psi} = \frac{\partial F}{\partial w} \frac{\partial w}{\partial \psi}.$$

# Dynamic regularization

**Outer iteration**: for $l = 0, 1, \ldots$ do

$$\left( \sum_{j=1}^{s} \hat{J}_j^T \hat{J}_j \right) \delta\psi = -\sum_{j=1}^{s} \hat{J}_j^T \left( F_j(w(\psi_l)) - b_j \right),$$

$$\psi_{l+1} = \psi_l + \tau \delta\psi$$

**Inner iteration** (for a singular system): a fixed number of preconditioned gradient descent (**PSD** or **PLSD**) or conjugate gradient **PCG** iterations.

Use discrete Laplacian as preconditioner for $\delta\psi$.

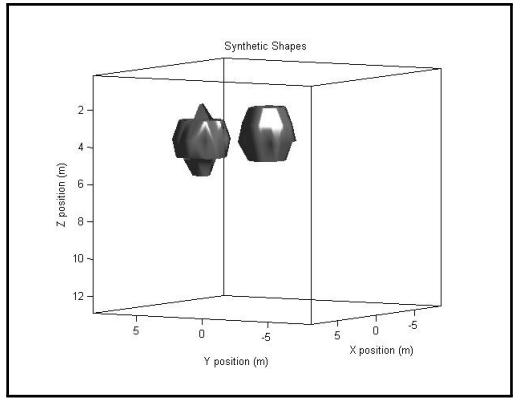
# Experiments

Setups and RESINVM3D from [Pidilsecky, Haber & Knight, '07]

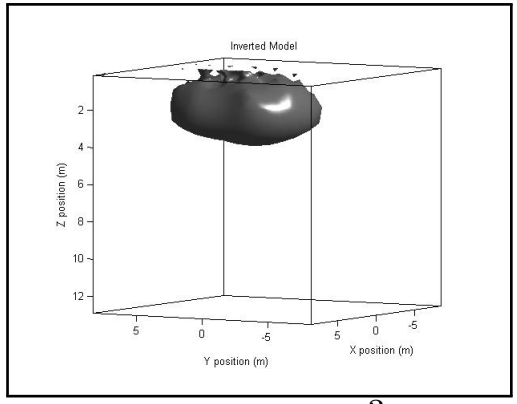
$$\nabla \cdot (e^w \boldsymbol{\nabla} u_j) = q_j, \quad j = 1, \ldots, s,$$
$$\frac{\partial u_j}{\partial \nu}|_{\partial\Omega} = 0,$$

discretized on a staggered grid

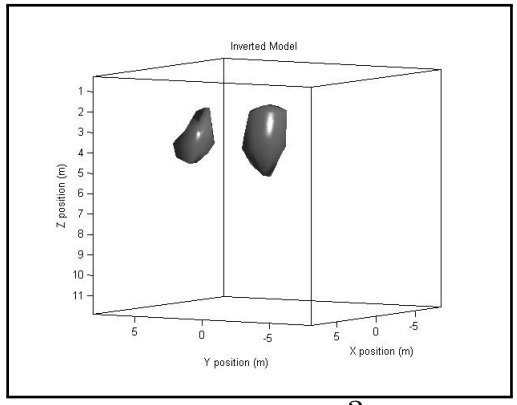
1. Surface electrodes, $w_I = -2.3, w_{II} = -5.3$, $s = 41$, $3\%$ noise
2. Subsurface electrodes, $3\%$ noise
3. Subsurface electrodes and receivers, $1\%$ noise



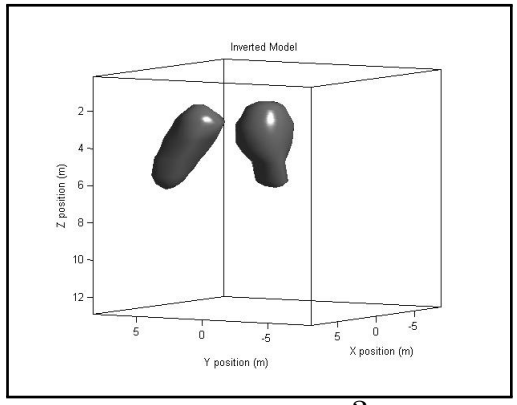

(t) Synthetic model.




(u) $L_2$-method, $32^3$ grid.

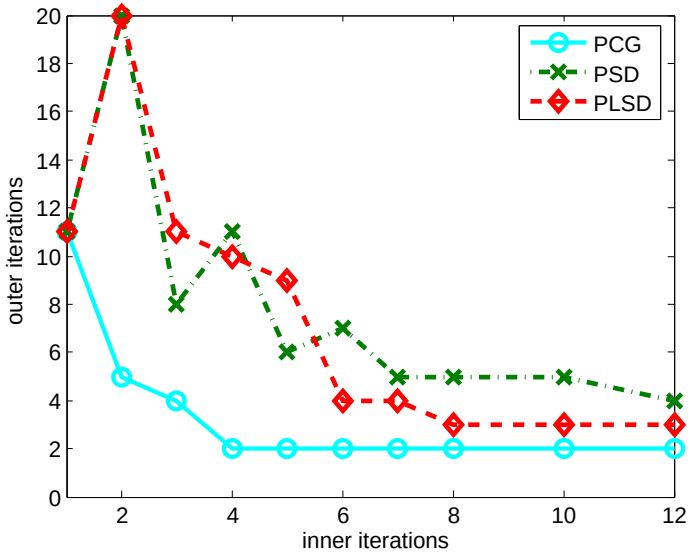



(v) Level set, $16^3$ grid.

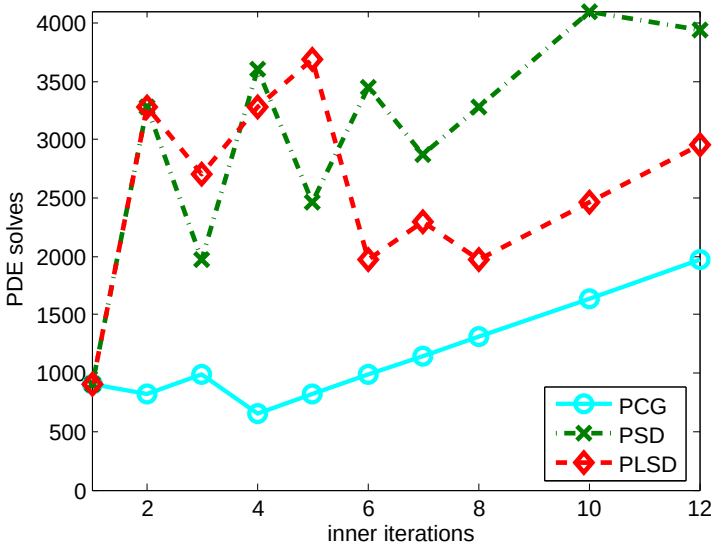



(w) Level set, $32^3$ grid.

# PSD, PCG and PLSD




(x) Outer iterations required




(y) Work units

Inner iteration counts and number of PDE solves = $s*2*iters_{in}*iters_{out}$.
For $iters_{in} = 1$ all three methods coincide.

# Outline

- Motivation
- Linear positive definite systems
- Chaos
- Regularization effects of gradient descent / artificial time
- Conclusions

# Conclusions

- **Faster gradient descent** methods yield **chaotic** dynamical systems.
- For unconstrained quadratic both one-step and two-step **gradient descent** methods can converge significantly faster than **steepest descent** (although normally not as fast as CG).
- Greedy algorithms SD and OM are inherently slow.
- There is still room for a better convergence theory for the **faster gradient descent** variants.

# Conclusions

- Occasionally some advantage can be gained by using LSD or some other faster method rather than SD as a smoother or regularizer.

- LSD or HLSD yield worthwhile advantage over SD when more than, say, 12 SD steps are required for an essentially quadratic minimization.

- Overall, the practical gains from LSD or HLSD do not appear to be huge, although they may be significant.

# References


- U. Ascher, K. van den Doel, H. Huang and B. Svaiter, “Gradient descent and fast artificial time integration”, M2AN (2009).

- K. van den Doel and U. Ascher, “The chaotic nature of faster gradient descent methods”, 2010.

- H. Huang and U. Ascher, “Faster gradient descent and the efficient recovery of images”, 2008.