


Math 4997-1

Lecture 9: Solvers, Conjugate gradient method, and Blazeterative

Patrick Diehl 

<https://www.cct.lsu.edu/~pdiehl/teaching/2020/4997/>

This work is licensed under a Creative Commons "Attribution-NonCommercial-NoDerivatives 4.0 International" license.



Reminder

Solving linear equation systems

Conjugate gradient method

The method of the steepest decent

Blaze Iterative

Summary

References

Reminder

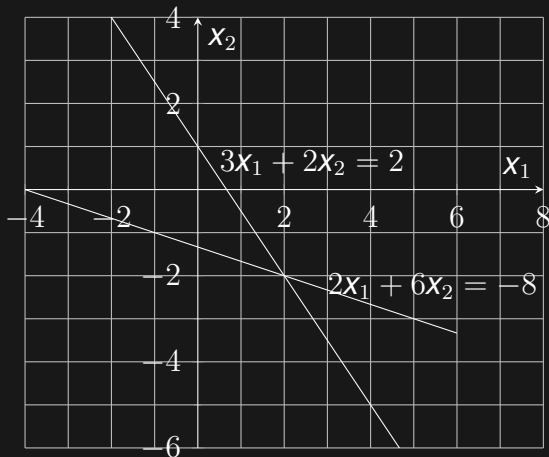
Lecture 9

What you should know from last lecture

- ▶ Vectors and matrices
- ▶ How to use Blaze for matrix and vector operations
- ▶ How to compile a program using a external library

Solving linear equation systems

Illustration of the linear system



Conjugate gradient method

Conjugate gradient method

Properties:

- ▶ Most popular iterative method for solving large systems of linear equations
- ▶ Developed by Hestenes and Stiefel in 1952 [3]
- ▶ Solves linear equation systems $\mathbf{Ax} = \mathbf{b}$
- ▶ Each iteration does one matrix-vector multiplication and some computation of inner products

Matrix

- ▶ Symmetry $\mathbf{A}^T = \mathbf{A}$
- ▶ Positive-definite $\mathbf{x}^T \mathbf{Ax} > 0, \forall \mathbf{x} > 0$

More details about iterative methods [2].

The quadratic form

Let us define the problem as a matrix:

$$\mathbf{Ax} = \mathbf{b}$$

with

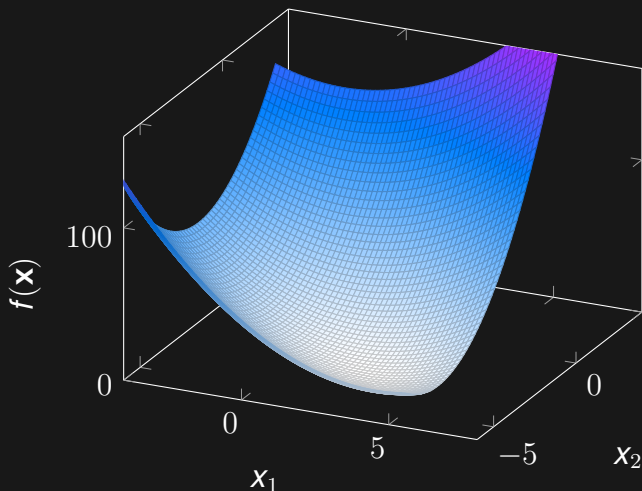
$$\mathbf{A} = \begin{pmatrix} 3 & 2 \\ 2 & 6 \end{pmatrix}, \mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \text{ and } \mathbf{b} = \begin{pmatrix} 2 \\ -8 \end{pmatrix}.$$

Instead of solving $\mathbf{Ax} = \mathbf{b}$, the quadratic form, which is a function of \mathbf{x} can be

$$f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^T\mathbf{Ax} - \mathbf{b}^T\mathbf{x} + c$$

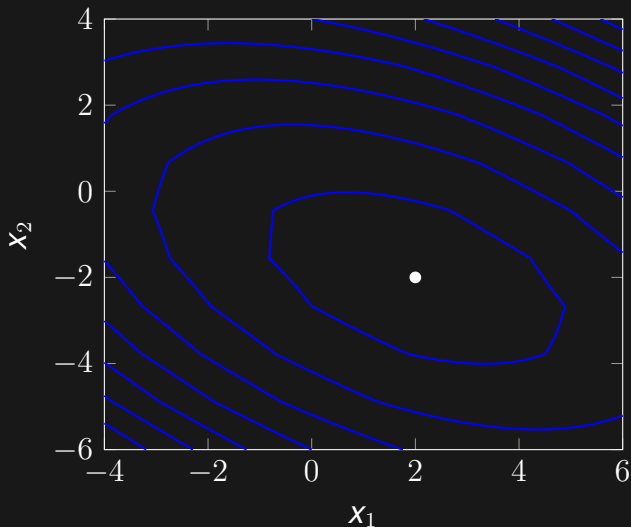
can be minimized to find the solution \mathbf{x} .

Plot of the quadratic form $f(\mathbf{x})$



Finding the minimal point of \mathbf{x} corresponds to the solution of $\mathbf{Ax} = \mathbf{b}$.

Contour plot of the quadratic form $f(\mathbf{x})$



Gradient of the quadratic form

Definition of the gradient:

$$f'(\mathbf{x}) = \begin{pmatrix} \frac{\partial}{\partial x_1} f(\mathbf{x}) \\ \frac{\partial}{\partial x_2} f(\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_n} f(\mathbf{x}) \end{pmatrix}$$

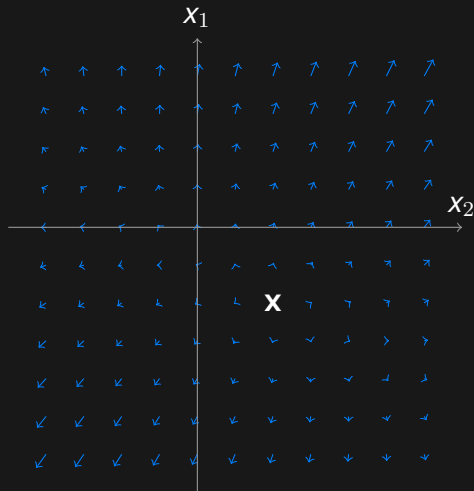
Applying a little bit of maths:

$$f'(\mathbf{x}) = \frac{1}{2} \mathbf{A}^T \mathbf{x} + \frac{1}{2} \mathbf{A} \mathbf{x} - \mathbf{b}$$

and for a symmetric matrix \mathbf{A} , we get

$$f'(\mathbf{x}) = \mathbf{A} \mathbf{x} - \mathbf{b}$$

Gradient field



Since the gradient at the solution \mathbf{x} is zero, we can set $f'(\mathbf{x})$ to zero to minimize $f(\mathbf{x})$.

The method of the steepest decent

- ▶ We chose an random point \mathbf{x}_0
- ▶ and slide down to the bottom of the quadratic form $f(\mathbf{x})$
- ▶ by taking a series of steps $\mathbf{x}_1, \mathbf{x}_2, \dots$
- ▶ Each step we go to the direction which f decreases most which is the opposite of $f'(\mathbf{x}_i)$ which is

$$-f'(\mathbf{x}_i) = \mathbf{b} - \mathbf{A}\mathbf{x}_i$$

The method of the steepest decent

Error

$$\mathbf{e}_i = \mathbf{x}_i - \mathbf{x}$$

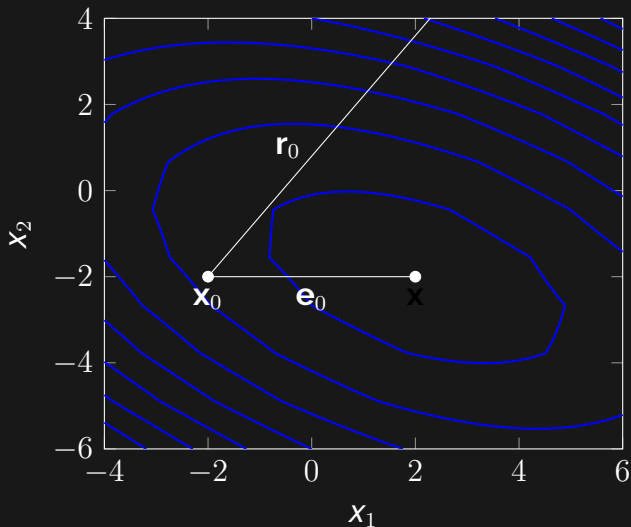
Defines how far way we are from the exact solution at iteration i .

Residual

$$\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i = -\mathbf{f}'(\mathbf{x}_i)$$

Defines how far away we are from the correct value for \mathbf{b} in iteration i .

Visualization of the residual and error



How far to go along the residual vector?

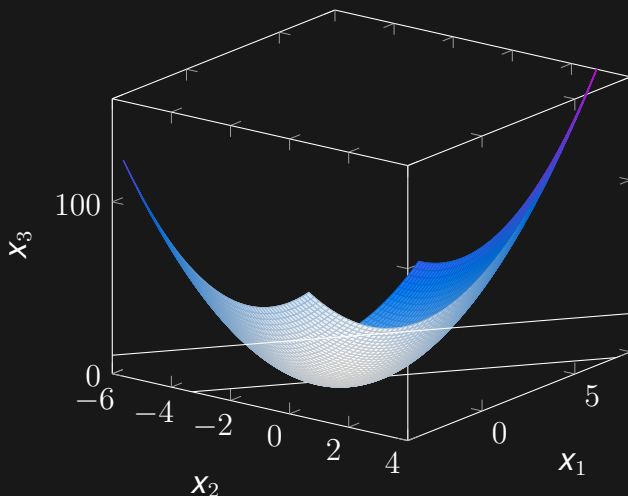
Line search

- ▶ We look at a starting point $\mathbf{x}_0 = [-2, -2]^T$
- ▶ from this point, we go along the direction of the steepest decent

$$\mathbf{x}_1 = \mathbf{x}_0 + \alpha \mathbf{r}_0$$

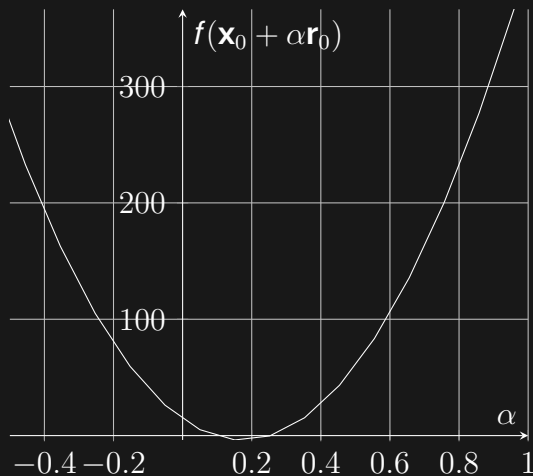
How large to chose α ?

Two surfaces



We need to find the point on the intersection of the two surfaces which minimizes f .

Parabola by the intersection of the two surfaces



The minimum of this function is as $\frac{d}{d\alpha} f(\mathbf{x}_0 + \alpha \mathbf{r}_0) = 0$.

How to determine α ?

Applying the chain rule: $\frac{d}{d\alpha}f(\mathbf{x}_0 + \alpha\mathbf{r}_0) = f'(\mathbf{x}_0 + \alpha\mathbf{r}_0)^T \mathbf{r}_0$.

This expression is zero, if the two vectors are orthogonal.

$$\mathbf{r}_1^T \mathbf{r}_0 = 0$$

$$(-\mathbf{A}\mathbf{x}_1)^T \mathbf{r}_0 = 0$$

$$(-\mathbf{A}(\mathbf{x}_0 + \alpha\mathbf{r}_0))^T \mathbf{r}_0 = 0$$

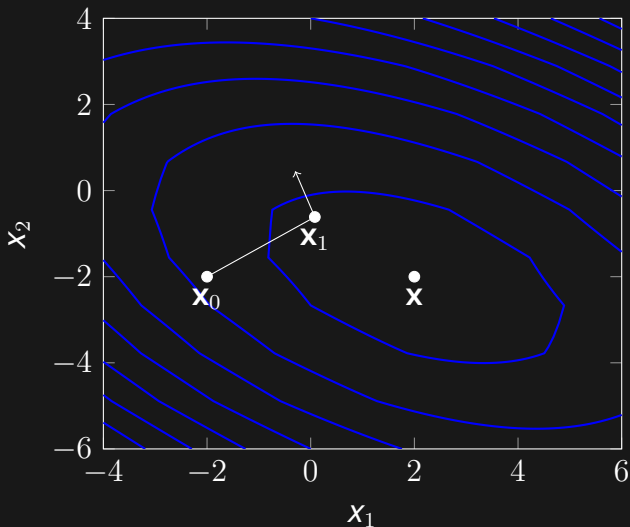
$$(\mathbf{b} - \mathbf{A}\mathbf{x}_0)^T \mathbf{r}_0 - \alpha(\mathbf{A}\mathbf{r}_0)^T \mathbf{r}_0 = 0$$

$$(\mathbf{b} - \mathbf{A}\mathbf{x}_0)^T \mathbf{r}_0 = \alpha(\mathbf{A}\mathbf{r}_0)^T \mathbf{r}_0$$

$$\mathbf{r}_0^T \mathbf{r}_0 = \alpha \mathbf{r}_0^T (\mathbf{A}\mathbf{r}_0)$$

$$\alpha = \frac{\mathbf{r}_0^T \mathbf{r}_0}{\mathbf{r}_0^T \mathbf{A}\mathbf{r}_0}$$

Visualization of gradient of the previous step

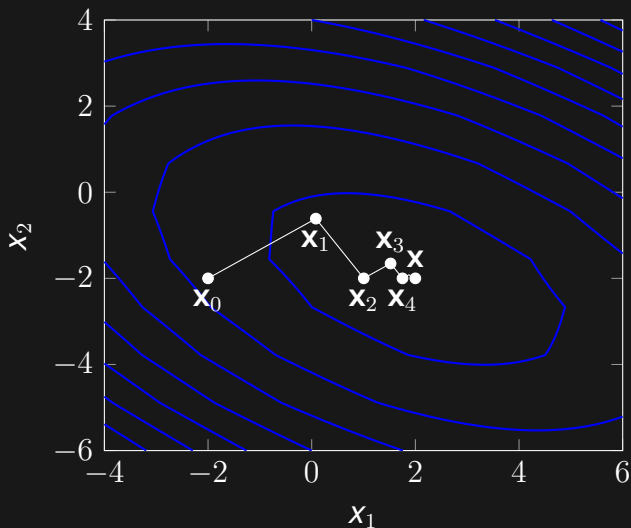


The gradient at \mathbf{x}_1 is orthogonal to \mathbf{x}_0 .

Algorithm

1. $\mathbf{r}_0 = \mathbf{b} - \mathbf{A}\mathbf{x}_0$
2. If $|\mathbf{r}_0| < \epsilon$ return \mathbf{x}_0
3. $\mathbf{r}_i = \mathbf{b} - \mathbf{A}\mathbf{x}_i$
4. $\alpha_i = \frac{\mathbf{r}_0^T \mathbf{r}_0}{\mathbf{r}_0^T \mathbf{A} \mathbf{r}_0}$
5. $\mathbf{x}_{i+1} = \mathbf{x}_i + \alpha_i \mathbf{r}_i$
6. If $|\mathbf{r}_i| < \epsilon$ return \mathbf{x}_i
7. Go to (3)

Visualization of the line search



The solution after five steps $\mathbf{x}_5 = [1.93832964, -2.]^T$.

Blaze Iterative

About BlazeIterative¹

This is a set of iterative linear system solvers intended for use with the Blaze library, a high-performance C++ linear algebra library. The API is currently based on a tag-dispatch system to choose a particular algorithm.

Usage

#Install

```
tar -xvf blaze_iterative.gz  
cd blaze_iterative  
cp -r ./blaze_iterative /home/patrick/
```

#Compile

```
g++ -I/home/diehlpk/blaze  
    -I/home/patrick/blaze_iterative BlazeTest.cpp
```

¹ <https://github.com/STELLAR-GROUP/BlazeIterative>

Conjugate gradient example

```
#include "BlazeIterative.hpp"

using namespace blaze;
using namespace blaze::iterative;

std::size_t N = 10;
DynamicMatrix<double,false> A(N,N, 0.0);
DynamicVector<double> b(N, 0.0);
DynamicVector<double> x1(N, 0.);

//Initialize the matrix

// Solve the system
ConjugateGradientTag tag;
auto x2 = solve(A,b,tag);
```

Available algorithms

Solvers

- ▶ Conjugate Gradient
- ▶ Preconditioned CG
- ▶ BiCGSTAB
- ▶ Generalized minimal residual method (GMRES),

Eigenvalues

- ▶ Lanczos

More details about solvers [1].

Summary

Summary

After this lecture, you should know

- ▶ Linear equation systems
- ▶ Conjugate gradient method
- ▶ Blazeterative

Acknowledgment

- ▶ The very nice example for the introduction of the conjugate gradient method was adapted from:
Shewchuk, Jonathan Richard. "An introduction to the conjugate gradient method without the agonizing pain." (1994).

References

References I

- [1] Richard Barrett, Michael W Berry, Tony F Chan, James Demmel, June Donato, Jack Dongarra, Victor Eijkhout, Roldan Pozo, Charles Romine, and Henk Van der Vorst.
Templates for the solution of linear systems: building blocks for iterative methods, volume 43.
Siam, 1994.
- [2] William L Briggs, Steve F McCormick, et al.
A multigrid tutorial, volume 72.
Siam, 2000.
- [3] Magnus Rudolph Hestenes and Eduard Stiefel.
Methods of conjugate gradients for solving linear systems, volume 49.
NBS Washington, DC, 1952.