

N -dimensional nonlinear equations

N -d solve



N -d optimization

Solving Nonlinear Equations

What is the goal here?

$$f \left(\begin{matrix} x_1 \\ \vdots \\ x_n \end{matrix} \right) = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} = \begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_n(x_1, \dots, x_n) \end{bmatrix}$$

f_n

Newton's method

What does Newton's method look like in n dimensions?

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \dots & \frac{\partial f_1}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_n}{\partial x_1} & \dots & \frac{\partial f_n}{\partial x_n} \end{bmatrix}$$

\downarrow

$$f(x_0 + s) \approx$$
$$= f(x_0) + J s$$

\leftarrow

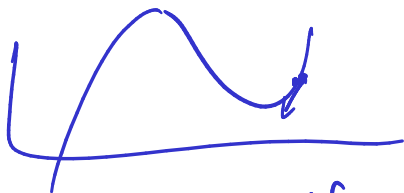
$$J s = -f(x_0)$$
$$s = -J^{-1} f(x_0)$$
$$x_1 = x_0 + s$$

Newton: Example

Set up Newton's method to find a root of

$$\mathbf{f}(x, y) = \begin{pmatrix} x + 2y - 2 \\ x^2 + 4y^2 - 4 \end{pmatrix}.$$

Demo: Newton's method in n dimensions



$$x_k = x_{k-1} - J_{x_{k-1}}^{-1} f(x_{k-1})$$

$$J = \begin{bmatrix} \frac{df_1}{dx} & \frac{df_1}{dy} \\ \frac{df_2}{dx} & \frac{df_2}{dy} \end{bmatrix}$$
$$J = \begin{bmatrix} 1 & 2 \\ 2x & 8y \end{bmatrix}$$

Secant in n dimensions?

What would the secant method look like in n dimensions?

$$1D: \quad x_{k+1} = x_k - \frac{f(x_k)}{\text{estimate of derivative at } x_k}$$

$$nD \quad = x_k - \frac{f(x_k)}{f'(x_k) - f(x_{k-1})}$$

$$J \Rightarrow x_{k+1} = x_k - J^{-1} f(x_k) \frac{1}{x_k - x_{k-1}}$$

$$\bar{J}_{x_k} - \bar{J}_{x_{k-1}} = f(x_k) - f(x_{k-1})$$

or

unknown

Broyden's method

minimize norm of $\bar{J}_{k+1} - \bar{J}_k$

Solving: - approximate function
with a linear expansion
- find zero of linear fun

Optimize: - approximate with quadratic
- minimize quadratic

Outline

Python, Numpy, and Matplotlib
Making Models with Polynomials
Making Models with Monte Carlo
Error, Accuracy and Convergence
Floating Point
Modeling the World with Arrays
 The World in a Vector
 What can Matrices Do?
 Graphs
 Sparsity
Norms and Errors
The 'Undo' Button for Linear Operations: LU
Repeating Linear Operations:
Eigenvalues and Steady States
Eigenvalues: Applications

Approximate Undo: SVD and Least Squares
SVD: Applications
 Solving Funny-Shaped Linear Systems
 Data Fitting
 Norms and Condition Numbers
 Low-Rank Approximation
Interpolation
Iteration and Convergence
Solving One Equation
Solving Many Equations
Finding the Best: Optimization in 1D
Optimization in n Dimensions

Optimization: What could go wrong?

What are some potential problems in optimization?

Optimization: What is a solution?

How can we tell that we have a (at least local) minimum? (Remember calculus!)

$$f'(x) = 0$$

Newton's Method

Let's steal the idea from Newton's method for equation solving:
Build a simple version of f and minimize that.

solve $g'(x) = f'(x) = 0$

apply Newton to $g'(x)$

$$x_k = x_{k-1} - \frac{g'(x_{k-1})}{g''(x_{k-1})}$$

$$= x_{k-1} - \frac{f'(x_{k-1})}{f''(x_{k-1})}$$

Demo: Newton's method in 1D

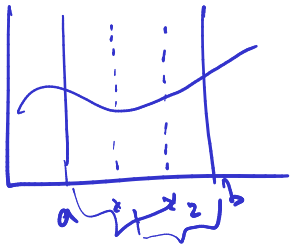
In-class activity: Optimization Methods

Golden Section Search

Would like a method like bisection, but for optimization.

In general: No invariant that can be preserved.

Need *extra assumption*.

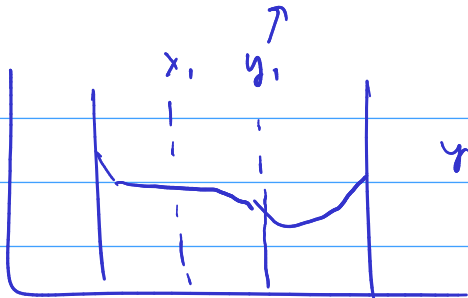


$$g(x) = x + a$$

$$x + a > 0$$

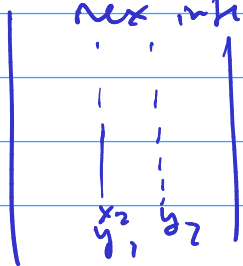
x^* - solution

f is unimodal



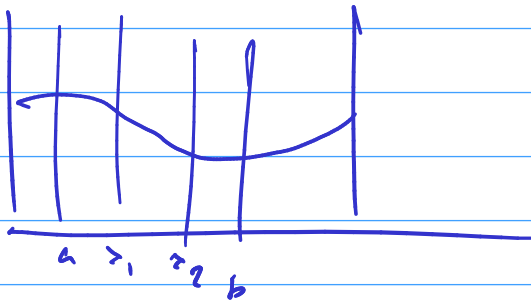
$$y = f(x) = \frac{1 + \sqrt{5}}{2}$$

next interval



if $f(x_1) > f(x_2) \rightarrow (x_1, b)$

if $f(x_1) \leq f(x_2) \rightarrow (a, x_2)$



Demo: Golden Section Search Proportions

Outline

Python, Numpy, and Matplotlib
Making Models with Polynomials
Making Models with Monte Carlo
Error, Accuracy and Convergence
Floating Point
Modeling the World with Arrays
 The World in a Vector
 What can Matrices Do?
 Graphs
 Sparsity
Norms and Errors
The 'Undo' Button for Linear Operations: LU
Repeating Linear Operations:
Eigenvalues and Steady States
Eigenvalues: Applications

Approximate Undo: SVD and Least Squares
SVD: Applications
 Solving Funny-Shaped Linear Systems
 Data Fitting
 Norms and Condition Numbers
 Low-Rank Approximation
Interpolation
Iteration and Convergence
Solving One Equation
Solving Many Equations
Finding the Best: Optimization in 1D
Optimization in n Dimensions

Optimization in n dimensions: What is a solution?

How can we tell that we have a (at least local) minimum? (Remember calculus!)

$\nabla f = 0 \rightarrow$ critical point

$H_f =$

$$\begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \dots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \vdots & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{bmatrix}$$

positive definite

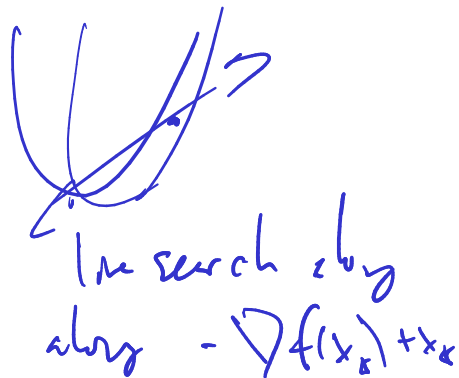
local minimum



Steepest Descent

Given a scalar function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ at a point x , which way is down?

x_0 - starting guess
- $\nabla f(x_0)$



CS 450 - Numerical Algorithms

CS 554 - Parallel Numerical Algorithms

CS 555

CS 598 - Andrew

Fast integral methods