# newton's method and optimization

Luke Olson

Department of Computer Science
University of Illinois at Urbana-Champaign



## semester plan

| | |
|---|---|
| Tu Nov 10 | Least-squares and error |
| Th Nov 12 | Case Study: Cancer Analysis |
| Tu Nov 17 | Building a basis for approximation (interpolation) |
| Th Nov 19 | non-linear Least-squares 1D: Newton |
| Tu Dec 01 | **non-linear Least-squares ND: Newton** |
| Th Dec 03 | Steepest Decent |
| Tu Dec 08 | Elements of Simulation + Review |

Friday December 11 – Tuesday December 15 Final Exam (computerized facility)

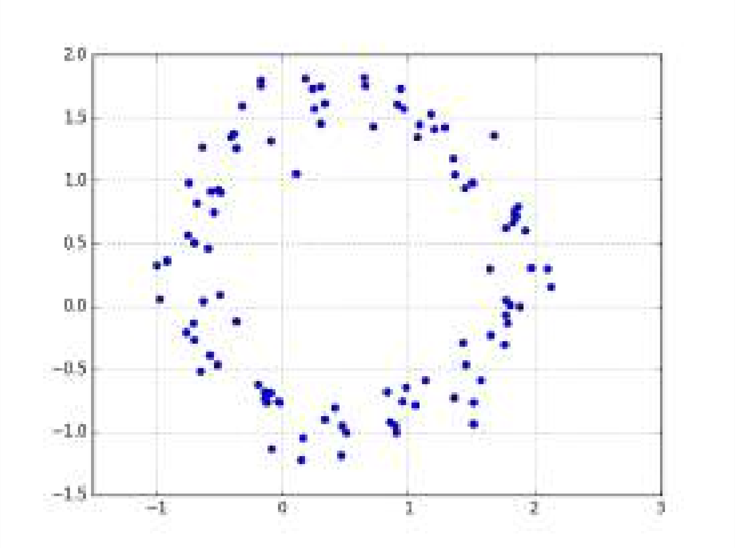
# objectives

- Write a nonlinear least-squares problem with many parameters
- Introduce Newton's method for $n$-dimensional optimization
- Build some intuition about minima



# fitting a circle to data

Consider the following data points $(x_i, y_i)$:




It appears they can be approximated by a circle. How do we find which one approximates it best?

# fitting a circle to data

What information is required to uniquely determine a circle? 3 numbers are needed:

- $x_0$, the x-coordinate of the center
- $y_0$, the y-coordinate of the center
- $r$, the radius of the circle.
- Equation: $(x - x_0)^2 + (y - y_0)^2 = r^2$

Unlike the sine function we saw before the break, we need to determine 3 parameters, not just one. We must minimize the residual:

$$R(x_0, y_0, r) = \sum_{i=1}^{n} \left((x_i - x_0)^2 + (y_i - y_0)^2 - r^2\right)^2$$

Do you remember how to minimize a function of several variables?



# minimization

A necessary (but not sufficient) condition for a point $(x^*, y^*, z^*)$ to be a minimum of a function $F(x, y, z)$ is that the gradient of $F$ be equal to zero at that point.

$$\nabla F = \left[\frac{\partial F}{\partial x}, \frac{\partial F}{\partial y}, \frac{\partial F}{\partial z}\right]^T$$

$\nabla F$ is a *vector*, and all components must equal zero for a minimum to occur (this does not guarantee a minimum however!).

Note the similarity between this and a function of 1 variable, where the first derivate must be zero at a minimum.

# gradient of residual

Remember our formula for the residual:

$$R(x_0, y_0, r) = \sum_{i=1}^{n} \left((x_i - x_0)^2 + (y_i - y_0)^2 - r^2\right)^2$$

Important: The variables for this function are $x_0$, $y_0$, and $r$ because we don't know them. The data $(x_i, y_i)$ is fixed (known).

The gradient is then:

$$\left[\frac{\partial R}{\partial x_0}, \frac{\partial R}{\partial y_0}, \frac{\partial R}{\partial r}\right]^T$$



# gradient of residual

Here is the gradient of the residul in all its glory:

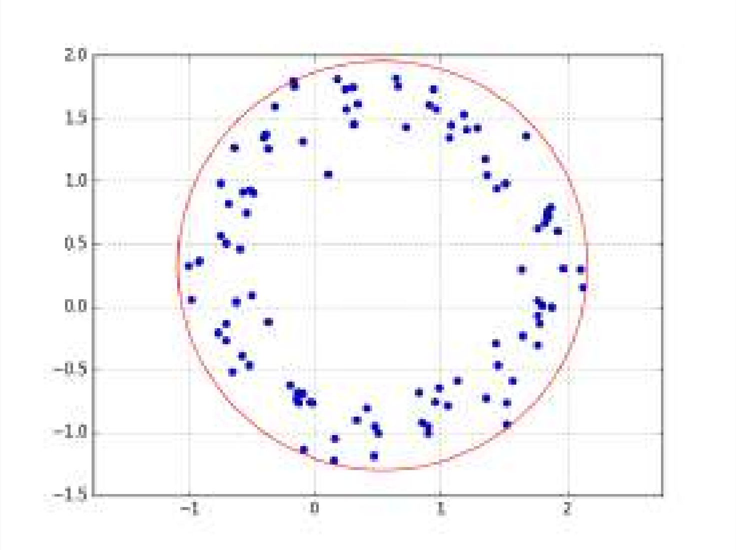
$$\begin{bmatrix} -4\sum_{i=1}^{n}\left[\left((x_i - x_0)^2 + (y_i - y_0)^2 - r^2\right)(x_i - x_0)\right] \\ -4\sum_{i=1}^{n}\left[\left((x_i - x_0)^2 + (y_i - y_0)^2 - r^2\right)(y_i - y_0)\right] \\ -4\sum_{i=1}^{n}\left[\left((x_i - x_0)^2 + (y_i - y_0)^2 - r^2\right) r\right] \end{bmatrix}$$

Each component of this vector must be equal to zero at a minimum. We can generalize Newton's method to higher dimensions in order to solve this iteratively.

We'll go over the details of the method in a bit, but let's see the highlights for solving this problem.

# newton's method

Just like 1-D Newton's method, we'll need an initial guess. Let's use the average *x* and *y* coordinates of all data points in order to guess where the center is. Let's choose the radius to coincide with the point farthest from this center:

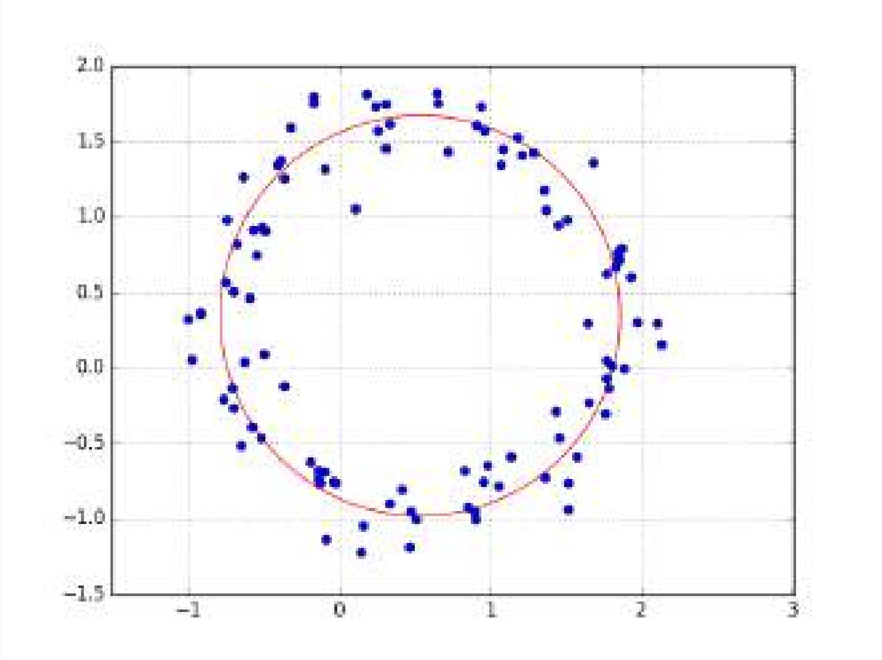



Not horrible...



# newton's method

After a handful of iterations of Newton's Method, we obtain the following approximate best fit:



# newton root-finding in 1-dimension

Recall that when applying Newton's method to 1-dimensional root-finding, we began with a linear approximation

$$f(x_k + \Delta x) \approx f(x_k) + f'(x_k)\Delta x$$

Here we define $\Delta x := x_{k+1} - x_k$. In root-finding, our goal is to find $\Delta x$ such that $f(x_k + \Delta x) = 0$. Therefore the new iterate $x_{k+1}$ at the $k$-th iteration of Newton's method is

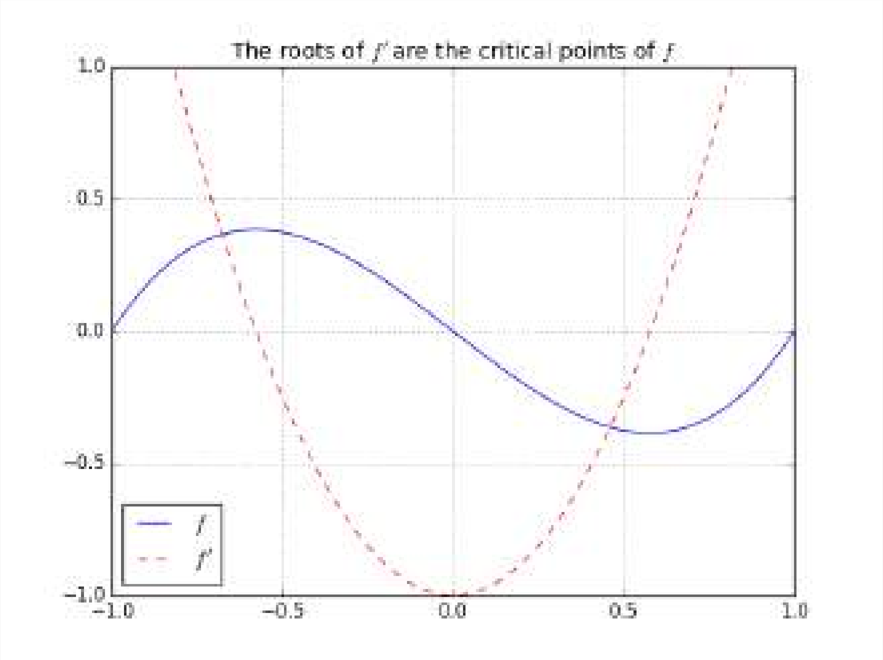
$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}}$$



# newton optimization in 1-dimension

Now consider Newton's method for 1-dimension optimization.

- For root-finding, we sought the zeros of $f(x)$.
- For optimization, we seek the zeros of $f'(x)$.



# newton optimization in 1-dimension

We will need more terms in our approximation, so let us form an approximation of second order

$$f(x_k + \Delta x) \approx f(x_k) + f'(x_k)\Delta x + f''(x_k)(\Delta x)^2$$

Next, take the partial derivatives of each side with respect to $\Delta x$, giving

$$f'(x_k + \Delta x) \approx f'(x_k) + f''(x_k)\Delta x$$

Our goal is $f'(x_k + \Delta x) = 0$, therefore the $k$-th iterate should be

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



# recall application to nonlinear least squares

From last class we had a non-linear least squares problem. We applied Newton's method to solve it.

$$r(k) = \sum_{i=1}^{m} (y_i - \sin(kt_i))^2$$

$$r'(k) = -2\sum_{i=1}^{m} t_i \cos(kt_i)(y_i - \sin(kt_i))$$

$$r''(k) = 2\sum_{i=1}^{m} t_i^2 \left[(y - \sin(kt_i))\sin(kt_i) + \cos^2(kt_i)\right]$$

Iteration:

$$k_{new} = k - \frac{r'(k)}{r''(k)}$$

# newton optimization in $n$-dimensions

- How can we generalize to an $n$-dimensional process?
- Need $n$-dimensional concept of a derivative, specifically
  - The Jacobian, $\nabla f(x)$
  - The Hessian, $Hf(x) := \nabla\nabla f(x)$

Then our second order approximation of a function can be written as

$$f(x_k + \Delta x) \approx f(x_k) + \nabla f(x_k)\Delta x + Hf(x_k)(\Delta x)^2$$

Again, taking the partials with respect to $\Delta x$ and setting the *LHS* to zero gives

$$\boxed{x_{k+1} = x_k - Hf^{-1}(x_k)\nabla f(x_k)}$$



# the jacobian

The Jacobian of a function, $\nabla f(x)$, contains all the first order derivative information about $f(x)$.

For a single function $f(x) = f(x_1, x_2, \ldots, x_n)$, the Jacobian is simply the gradient

$$\nabla f(x) = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_n}\right)$$

For example:

$$\begin{aligned} f(x, y, z) &= x^2 + 3xy + yz^3 \\ \nabla f(x, y, z) &= (2x + 3y,\ 3x + z^3,\ 3yz^2) \end{aligned}$$

# the hessian

Just as the Jacobian provides first-order derivative information, the Hessian provides all the second-order information

The Hessian of a function can be written out fully as

$$Hf(x) = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1 \partial x_1} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & & & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n \partial x_n} \end{bmatrix}$$

In a concise notation using element-wise notation

$$Hf_{i,j}(x) = \frac{\partial^2 f}{\partial x_i \partial x_j}$$



# the hessian

An example is a little more illuminating. Let us continue our example from before.

$$\begin{aligned} f(x, y, z) &= x^2 + 3xy + yz^3 \\ \nabla f(x, y, z) &= (2x + 3y,\ 3x + z^3,\ 3yz^2) \\ Hf(x, y, z) &= \begin{bmatrix} 2 & 3 & 0 \\ 3 & 0 & 3z^2 \\ 0 & 3z^2 & 6yz \end{bmatrix} \end{aligned}$$

# notes on newton's method for optimization

- The roots of $\nabla f$ correspond to the critical points of $f$
- But in optimization, we will be looking for a specific type of critical point (e.g. *minima* and *maxima*)
- $\nabla f = 0$ is only a necessary condition for optimization. We must check the second derivative to confirm the type of critical point.
- $x^*$ is a minima of $f$ if $\nabla f(x^*) = 0$ **and** $Hf(x^*) > 0$ (i.e. positive definite).
- Similarly, for $x^*$ to be a maxima, then we need $Hf(x^*) < 0$ (i.e. negative definite).



# notes on newton's method for optimization

- Newton's method is dependent on the initial condition used.
- Newton's method for optimization in $n$-dimensions requires the inversion of the Hessian and therefore can be computationally expensive for large $n$.