STA410 | Programming Portfolio Assignment 3

3.0 - Nonlinearity | $b=f(x)\not=Ax$

A more interesting problem than $Ax=b$ is the problem $g(x)=b$ for a ***nonlinear function*** g. Such ***nonlinear equations*** are a general version form of a wide set of problems. However, a most frequently encountered instance of the problem is

$$f'(z) = 0 \quad \text{ or more generally } \quad \underbrace{\nabla_z f(z) = \mathbf{0}}_{\text{multivariate form}}$$

because solutions to these equations are ***local minima*** or ***maxima*** of $$f$$ and ***model fitting*** can be framed as a subclass of this general ***optimization problem.***

Two important notes about these ***optimization problems*** can be made.

1. The ***derivative*** $f'(z)$ is just some other function, say $g(z)$, so outside of its useful interpretation as a derivative, $f'(z)$ may be treated just as any other function might be treated.

> E.g., the ***first-order Taylor series approximation*** of f' is simply
>
> $$f'(x) \approx f'(x_0) + (x-x_0)f''(x_0)$$
>
> as if f' had been replaced with some other function g.

2. ***Optimization solutions*** $f'(x^*)=0$ are found within regions of curvature of f, while ***roots*** $f(x_0)=0$ need not be.

- The behavior of a function near a ***solution*** to an ***optimization*** problem differs, generally speaking, from it's behavior near a ***root***.
- And the ***numerical precision*** in an ***optimization*** context to in general differs from the vanilla ***root-finding*** context.

> E.g., to the degree that a ***second order Taylor series approximation*** $g_{\tilde x}(x)$ of $f(x)$ is accurate
>
> $$f(x) \approx g_{\tilde x}(x) = f(\tilde x) + (x - \tilde x) f'(\tilde x) + f''(\tilde x)\frac{(x-\tilde x)^2}{2} $$
>
> changes in $g_{\tilde x}(x)$ are likely dominated by
>
> - the linear term $(\underbrace{x-\tilde x}_{\epsilon\text{machine}})f'(\tilde x) \longrightarrow$ if $\tilde x$ is near ***root*** x_0
> - the quadratic term $\frac{1}{2}(\underbrace{x-\tilde x}_{\sqrt{\epsilon\text{machine}}})^2f''(\tilde x) \longrightarrow$ if $\tilde x$ is near ***optimum*** $x^$, *since $f'(\tilde x \approx x^)\approx 0$.*
>
> I.e., $f(x) \approx g_{x^}(x)$ *evaluated near an* ***optimization problem solution*** *$x^$* only supports about half of the ***numerical accuracy***

> > i.e., the square root of the available precision $\epsilon_{machine}$, since the difference will be squared
>
> compared to evaluating $f(x) \approx g_{x_0}(x)$ near one of its ***roots*** x_0.

==========

3.0.1 Maximum Likelihood Estimates (MLEs)

The ***score function*** is the gradient of the ***log likelihood***

$$\nabla_\theta l(\theta') = \left(\frac{\partial l(\theta')}{\partial \theta_1}, \cdots, \frac{\partial l(\theta')}{\partial \theta_p} \right)^T \quad \text{ where } \quad l(\theta) = \log f(x|\theta) \overset{iid}{=} \log \prod_{i=1}^n f(x_i|\theta)$$

Maximum Likelihood Estimates (MLEs) come from the (***nonlinear***) ***score equation*** which sets the ***score function*** equal to $\mathbf{0}$. And for the ***true value*** of the parameter θ^{true}, the ***score function*** has expected value $\mathbf{0}$ (with respect to f_x the distribution of the data). So

$$\underbrace{\nabla_\theta l(\hat \theta) = \mathbf{0}}_{\text{score equation}} \quad \text{ and } \quad E_X !\underbrace{\left[\nabla_\theta l(\theta^{\text{true}})\right]}_{\text{score function}} != \mathbf{0}$$

> The expected value of the ***score function*** follows since
>
> \begin{align*} E \left[\nabla_\theta l(\theta)\right]
> = {} & \int \nabla_\theta l(\theta) f(x|\theta) dx \ = {} & \int \left(\frac{\partial l(\theta)}{\partial \theta_1}, \cdots, \frac{\partial l(\theta)}{\partial \theta_p} \right)^T f(x|\theta) dx \ = {} & \int \left(\frac{1}{f(x|\theta)}\frac{\partial f(x|\theta)}{\partial \theta_1}, \cdots, \frac{1}{f(x|\theta)}\frac{\partial f(x|\theta)}{\partial \theta_p} \right)^T f(x|\theta) dx \ = {} & \int \left(\frac{\partial f(x|\theta)}{\partial \theta_1}, \cdots, \frac{\partial f(x|\theta)}{\partial \theta_p} \right)^T dx \ = {} &\int \nabla_\theta f(x|\theta) dx \ = {} & \nabla_\theta \int f(x|\theta) dx = \nabla_\theta , 1 = \mathbf{0} \ \end{align*}

The ***Fisher information matrix*** $I(\theta^{\text{true}})$, or ***expected Fisher information matrix*** is the expected value of the ***outer product*** of the ***score function*** with itself and is equal to the expected value of the negative of the ***Hessian*** of the log likelihood (all with respect to f_x the distribution of the data), i.e., I
$$I(\theta^{\text{true}}) = E_X[\nabla_\theta l(\theta^{\text{true}})(\nabla_\theta l(\theta^{\text{true}})^T] = E_X[-H_{l(\theta)}(\theta^{\text{true}})]$$

The ***observed Fisher information*** is

$$\begin{align*} \hat{I(\theta)} = {} & -H_{l(\theta)}(\hat \theta) = \underbrace{-J(\nabla_\theta l)(\hat \theta) \approx - \sum_{i=1}^n J(\nabla_\theta log f(x_i|\theta))\big|_{\hat \theta}}_{\text{$J(\nabla_\theta l) (\hat \theta)$ }\textit{Jacobian}\text{ of }\textit{score function} \text{ evaluated at } \hat \theta} \ \text{ or } \quad \hat{I(\theta)} \approx {} & \sum_{i=1}^n \nabla_\theta log f(x_i|\theta)\big|_{\hat \theta}\left(\nabla_\theta log f(x_i| \theta)\big|_{\hat \theta}\right)^T \end{align*}$$

And the ***asymptotic distribution*** of the MLE is

$$ p(\hat \theta) \overset{n \rightarrow \infty}{\longrightarrow} N!\left(\theta^{\text{true}}, \Sigma = \frac{I(\theta^{\text{true}})^{-1}}{n}\right) \approx N!\left(\theta^{\text{true}}, \Sigma = \frac{\hat{I(\theta)}

{}^{-1}}{n} \approx \frac{I(\theta^{\text{true}})^{-1}}{n}\right)$$

where either the ***observed Fisher information*** or its approximation based on the ***outer product*** of the ***score function*** with itself may be used as plug in estimates for the ***expected Fisher information matrix*** $I(\theta^{\text{true}})$.

========

The Jacobian J

The ***Hessian*** $H_{f(z)}$ matrix of ***second order partial derivatives*** of $f(z)$ is (of course) distinct from the ***Jacobian*** J, which is a different matrix of ***first order partial derivatives*** for the ***multivariate*** $y = g(z)$ which maps $x \in {\rm I!R}^p$ to $y\in {\rm I!R}^q$

$$ [Jg(z)(z')]_{ij} = \frac{\partial g_i(z')}{\partial z_j}
\quad \text{ so } \quad Jg(z)(z') = \begin{array}{c}\overset{y_1}{\underset{y_q}{\Bigg \downarrow}}\end{array}
\overset{z_1 ; -!-!-!-!-!-!-!-!!\longrightarrow ; z_p}{\left[\begin{array}{ccc} \frac{\partial g_1(z')}{\partial z_1} &
\cdots & \frac{\partial g_1(z')}{\partial z_p}\ \vdots & \ddots & \vdots \ \frac{\partial g_q(z')}{\partial z_1} &
\cdots & \frac{\partial g_q(z')}{\partial z_p} \end{array} \right]} = \left[\begin{array}{c}(\nabla_z g_1
)^T(z')\\vdots \ (\nabla_z g_i)^T(z') \\vdots \ (\nabla_z g_p)^T(z') \end{array}\right]$$

where g_i is the i^{th} element of the multivariate output of $g(z)$.

> The ***first order Taylor Series approximiation*** for functions with both multivariate inputs and outputs can be expressed by replacing the ***gradient*** with the ***Jacobian***, i.e.,
>
> $$\underbrace{f(\theta) \approx f(\theta_0) + (\nabla_\theta f)^T(\theta_0)(\theta-\theta_0)}*{\text{when f has multivariate input and univariate output}} \quad \text{ generalizes to } \quad \underbrace{f(\theta) \approx f(\theta_0) + J f*\theta (\theta_0)(\theta-\theta_0)}_{\text{when f has both multivariate input } \textbf{and} \text{ output}} $$
>
> since the latter simply specifies a matrix multiplication which forms a vector whose i^{th} element is the ***first order Taylor Series approximiation*** for the i^{th} univariate output f_i of the multivariate output of f.

As noted previously, and now seen clearly from the definition of the ***Jacobian***, the ***Hessian*** is

$$\begin{align*}H_{f(z)}(z') = {} & J(\nabla_z f)(z') \ = {} & \left[\frac{\partial}{\partial z_1}\nabla_z f(z') ;;\cdots;; \frac{\partial}{\partial z_j}\nabla_z f(z') ;;\cdots;; \frac{\partial}{\partial z_p}\nabla_z f(z') \right] \end{align*}$$

> Another relationship between the ***Hessian*** and the ***Jacobian*** exists in the context of a ***least squares*** problem; namely, applying a ***first order Taylor series approximation*** around optimal $\theta^$ *within the* ***nonlinear least squares*** *problem* \begin{align}\min_\theta \frac{||y - f_\theta(x)||2^2}{2} \approx {} & \min\theta \frac{||y - \big(\overbrace{f_{\theta^}(x)}^{f_x(\theta^)}+\overbrace{J(f_{\theta}(x))(\theta^)}^{J_{f_x(\theta)}(\theta^)}(\theta-\theta^)\big)||_2^2}{2} \ = {} & \min \underbrace{ \left(y - \left(f_x(\theta^)+J_{f_x(\theta)}(\theta^)(\theta-\theta^)\right) \right)^T\frac{1}{2}\left(y - \left(f_x(\theta^)+J_{f_x(\theta)}(\theta^)(\theta-\theta^)\right) \right) }_{g(\theta)} \end{align}
>
> means the ***Hessian*** of the ***least squares objective function*** to the first order approximation of $g(\theta)$ is the ***inner product*** of the ***Jacobian*** with itself $$H_{g(\theta)} = \left(J_{f_x(\theta)}

(\theta^)\right)^T\left(J_{f_x(\theta)}(\theta^)\right)$$

> The ***inner product*** of the ***Jacobian*** is the ***outer products*** of ***gradients***, so the result here is lines up with the ***outer product*** definition of ***Fisher information*** (based on summing over the observations).
>
> The negative sign on the ***Hessian*** for ***Fisher information*** is simply because ***Fisher information*** is ***positive definite*** while the ***hessian*** is ***negative definite*** since ***log likelihood (MLE) optimization*** is a ***maximization*** problem
>
> - $H_{l(\theta)}(\hat \theta)$ at ***local maxima*** $\hat \theta$ is ***negative definite***
>
> whereas, the ***least squares optimization*** is a ***minimization*** problem
>
> - $H_{g(\theta)}(\theta^)$ at ***local minima*** $\theta^$ is ***positive definite***,

- This is the ***nonlinear least squares*** problem generalizing the ***least squares*** problem from Section 1.1.2 and Section 1.2.2. An ***iterative*** approach to solving this problem, building upon the "Iterative Methods" (Section 1.2) is continued in the "Gauss-Newton" section (Section 3.4.0) below.

=======

3.0.2 (Bayesian) Variational Inference

Approximate Bayesian analysis may proceed on the basis of

$$ \begin{align*} \log p(y) = {} & \int \log p(y) q(\theta) d\theta \ = {} & \int \log \frac{p(\theta,y)}{p(\theta|y)} \frac{q(\theta)}{q(\theta)} q(\theta) d\theta = \int \log \frac{p(y|\theta)p(\theta)}{p(\theta|y)} \frac{q(\theta)} {q(\theta)} q(\theta) d\theta \ = {} & \underbrace{\int \log p(y|\theta) q(\theta) d\theta }{E{q(\theta)}[\log p(y|\theta)]} + KL[, q(\theta) , || , p(\theta|y) ,] - KL[, q(\theta) , || , p(\theta) ,] \end{align*} $$

> which can be seen to define ***Kullback-Leibler (KL) divergence***; namely, e.g., $$KL[, q(\theta) , || , p(\theta) ,] = \int \frac{q(\theta)}{p(\theta)} q(\theta) d \theta$$

since optimally approximating the posterior $p(\theta|y)$ with $q(\theta)$, i.e.,

$$\min_q KL[, q(\theta) , || , p(\theta|y) ,] \quad \underset{\text{(since KL terms are nonnegative)}}{\text{ is equivalent to }} \quad \max_q \text{ELBO}(q) \quad \left(\text{or } \min_q -\text{ELBO}(q)\right)$$

i.e., maximizing the ***Evidence Lower BOund***

$$\log p(y) \geq \overbrace{E_{q(\theta)}[\log p(y|\theta)] - KL[, q(\theta) , || , p(\theta) ,]}^{\text{ELBO}(q)}$$

Approximating the ***posterior*** in this manner is known as ***variational inference*** and is accomplished by solving the ***nonlinear equation***

$$\nabla_\phi\text{ELBO}(q_\phi(\theta)) = \mathbf{0}$$

> - The $\text{ELBO}(q)$ may be equivalently defined in terms of only expectations (and no KL-terms) since, using ***Jensen's inequality***

> $ \begin{align*} \log p(y) = {} & \log \int p(y,\theta) \frac{q(\theta)}{q(\theta)} d\theta \quad \quad \longrightarrow &={}& \log \int p(y|\theta)\frac{p(\theta)}{q(\theta)} q(\theta) d\theta \ = {} & \log E_{q(\theta)}\left[p(y,\theta)\frac{1}{q(\theta)} \right] \quad ; \longrightarrow &={}& \log E_{q(\theta)}\left[p(y|\theta)\frac{p(\theta)}{q(\theta)} \right]\ \geq {} & E_{q(\theta)}\left[\log \frac{p(y,\theta)}{q(\theta)} \right] \quad \quad ;;; \longrightarrow &={} & E_{q(\theta)}\left[\log \left(p(y|\theta)\frac{p(\theta)}{q(\theta)}\right) \right] \ = {} & \underbrace{E_{q(\theta)}[\log p(y,\theta)] - E_{q(\theta)}[\log q(\theta)]}*{\text{ELBO}(q)} &={} & \underbrace{E*{q(\theta)}[\log p(y|\theta)] - KL[, q(\theta) , || , p(\theta) ,]}_{\text{ELBO}(q)} \end{align*} $
>
> - And the $\text{ELBO}(q)$ is often just derived directly from the desired Bayesian optimization
>
> $ \begin{align*} KL[, q(\theta) , || , p(\theta|y) ,] = {} & \int \log \frac{q(\theta)}{p(\theta|y)}q(\theta)d\theta\ = {} & \int \log \frac{q(\theta)p(y)}{p(y|\theta)p(\theta)}q(\theta) d\theta\ = {} & \log p(y) + \int \log \frac{q(\theta)}{p(\theta)}q(\theta) d\theta - \int \log p(y|\theta) q(\theta) d\theta \ = {} & \log p(y) + \underbrace{KL[, q(\theta) , || , p(\theta) ,] - E_{q(\theta)}[\log p(y|\theta)]}_{-\text{ELBO}(q)} \ \end{align*} $

==========

3.1 Nonlinear Root-Finding: $0=f(x)\not=Ax$

The nonlinear equation $f(x)=0$ characterizes an extremely general and useful class of problems, e.g., MLE or approximate Bayesian analysis.

Unfortunately, despite (or perhaps because of) this generality, there are often no analytic solutions to nonlinear equations $f(x)=0$. Thus, root-finding for nonlinear functions $f(x)$ often simply proceeds on the basis of ***iteratively*** improving numerical approximations.

As first noted during their initial introduction (in Section 1.2), the highest level considerations for ***iterative methods*** are:

- The "per step cost versus number of steps" tradeoff

  > e.g., as observed in the comparison of ***Newton's method*** versus the ***secant method*** (in Section 3.1.2 below)

- Convergence based on a ***stopping criterion*** using

  relative convergence

  $$\text{ }\quad \frac{|x^{(t+1)}-x^{(t)}|}{x^{(t)}} < \epsilon \quad \text{ or } \quad \frac{|x^{(t+1)}-x^{(t)}|}{x^{(t)} + \epsilon} <0 ; \text{ if } ; x^* \approx 0$$

  or ***absolute convergence***

  $$\quad |x^{(t+1)}-x^{(t)}| < \epsilon ; \text{ (which is of course scale dependent)}$$

  along with additional ***timeout*** contingencies based on cycling, lack of improvement, or simply the number of ***iterations*** K.

Initializations are also very important for ***iterative methods***.

- Starting near a (global) optimum is of course helpful.

  > E.g., with ***method of moments*** initialization.

- Trying multiple short trajectories with different initializations gives more powerful exploration than a single long-running trajectory

  > I.e., ***embarassingly simple parallelization***.

  and correctness of solutions can be supported by confirming similarity of results for differently initialized trajectories.

  ====

3.1.0 Bisection

Root-finding with ***bisection*** (for ***continuous*** functions) was introduced in the "Limits of Precision: Root-Finding with Bisection" (Section 0.2.1), but here are some generally instructive considerations characterizing ***bisection***:

0. *Accuracy*: root-finding exactness is constrained by the available numeric precision.

1. *Initialization*: zeros cannot be found without initializing $\text{sign}(f(a_0)) \not = \text{sign}(f(b_0))$

2. *Limitation*: a maximum of one zero in the initial interval $[a_0, b_0]$ can be found

3. *Convergence*: since the interval is halved at each step, the width of the interval at step t is

   $$b_t-a_t = 2^{-t}(b_0-a_0)$$

   so that for true root $f(x^)=0$ where $x^ \in [a_t, b_t]$

   $$\left|\frac{b_t+a_t}{2} - x^*\right| < \delta \quad \text{ when } \quad 2^{-(t+1)}(b_0-a_0) < \delta \quad \text{ or } \quad t > log_2\left(\frac{(b_0-a_0)}{\delta}\right) + 1$$

so to reduce δ by a factor of 10 (i.e., to reduce the error by one unit of precision) requires

$$t -1 > log_2\left(\frac{(b_0-a_0)}{\delta/10}\right) = log_2\left(\frac{(b_0-a_0)}{\delta}\right) + \log_2(10) $$

an about $\log_2(10) \approx 3.2$ more steps:

```
np.log2(10) # 3.321928094887362
```

> The ***bisection method*** is one (of many) ***braketing methods***. Another, generally faster, ***braketing method*** method is known as the ***Illinois method***.

- The ***Illinois method*** is a generally preferred variant of the ***Regula Falsi*** method which has an extremely long and interesting history.

- The ***Regula Falsi*** method works just like the ***secant method***, which is discussed as a discrete version of ***Newton's method below***; except, the sequential points are chosen to bracket the solution.

====

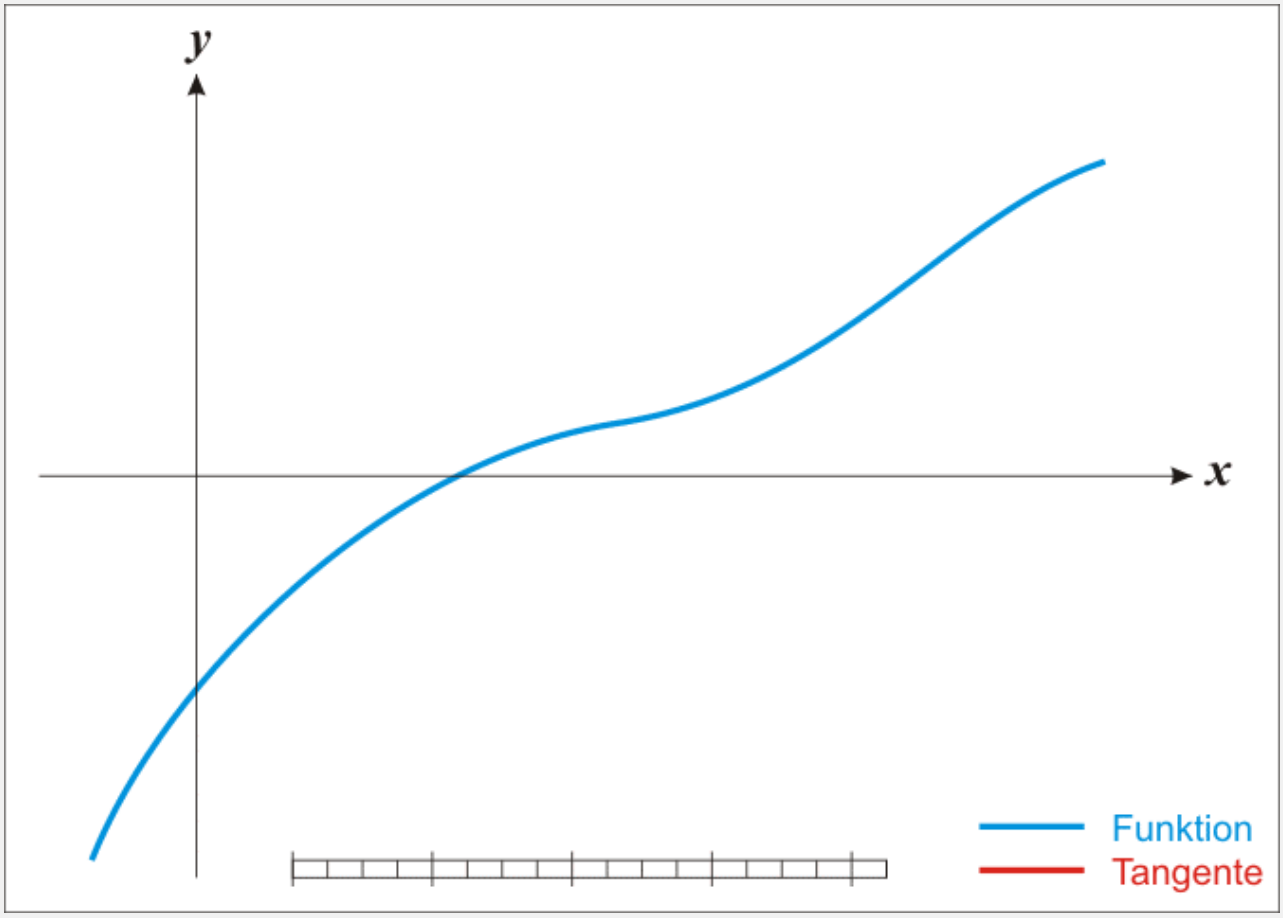
3.1.1 Newton's Method

Also known as ***Newton-Raphson iteration***, using a first-order Taylor series approximation of g' in the equation of interest

$$\overbrace{0 = g'(x^*)}^{\text{original problem}} \approx g'(x^{(t)}) + (x^*-x^{(t)})g''(x^{(t)}) =0$$

means so long as g'' exists (and is non-zero at $x^{(t)}$)

$$ x^* = x^{(t)} - \frac{g'(x^{(t)})}{g''(x^{(t)})} \quad \text{ which suggests the iterative update } \quad x^{(t+1)} = x^{(t)} + h^{(t)} \quad \text{ with } \quad h^{(t)}=- \frac{g'(x^{(t)})}{g''(x^{(t)})} $$

> Since $g''(x)$ is the (instantaneous) slope of g' at x, solving the the first order Taylor series approximation of g' for 0 simply means following the slope in the direction of the $y=0$ line until hitting it.
>
> From Wikipedia's Newton's Method page, considering $f(x)$ to be $g'(x)$ above:
>
> 
>

- The first question of Programming Portfolio Assignment 3 will involve implementing a ***univariate Newton's method***.

> While ***Newton's method*** is introduced here in the context of solving $g'(x^*) = 0$ *(and thus ends up depending on g''); however,* ***Newton's method*** *more generally simply a* ***root-finding*** *method based*

> *on a **first-order Taylor series approximation** of any function f (and not necessarily an approximation of the **derivative** of a function g'). This is why, e.g., wikipedia presents **Newton's method** without any attention to the $g'(x^) = 0$ problem.*

2. *Limitation*: initialization agnostic convergence is only guaranteed for some functions, e.g., ***twice differentiable convex functions (with roots)***.

3. *Convergence*: a ***second-order Taylor series approximation*** of g' set equal to 0 gives $ \begin{align*} & g'(x^) \approx \ & g'(x^{(t)}) + g''(x^{(t)})(x^-x^{(t)}) + \frac{1}{2}g'''(x^{(t)})(x^-x^{(t)})^2 = 0\ & x^{(t)}-\frac{g'(x^{(t)})}{g''(x^{(t)})} - x^ = (x^-x^{(t)})^2 \frac{g'''(x^{(t)})}{2g''(x^{(t)})}\ & \quad \quad \quad ;x^{(t+1)}-x^ = (x^-x^{(t)})^2 \frac{g'''(x^{(t)})}{2g''(x^{(t)})} \ & \frac{\hat \epsilon^{(t+1)}}{(\hat \epsilon^{(t)})^2} = \frac{g'''(x^{(t)})}{2g''(x^{(t)})} \ \Longrightarrow \quad & \lim_{t\rightarrow \infty} \frac{|\hat \epsilon^{(t+1)}|}{|\hat \epsilon^{(t)}|^\beta} = c \quad \text{ where } \beta=2 \text{ and } \underset{\text{assuming convergence}}{ c = \left|\frac{g'''(x^{})}{2g''(x^{})}\right|}\ \end{align}$

   which is the defintion of ***convergence of order*** $\beta=2$ (i.e., ***quadratic convergence***). So when a ***second-order Taylor series closely approximates*** g', the precision of the converging answer increases approximately doubles at each iteration.

   - E.g., $0.1^2 = 0.01$ and $0.01^2 = 0.0001$.

   > Since the midpoint of the interval for the ***bisection method*** need not necessarily get closer to the answer every single step, the error of ***bisection method*** does not formally converge to 0; rather, it is the bracketing interval width that is halved with each step. Letting $\tilde \epsilon$ be the bracketing interval width of the ***bisection method*** $$\lim_{n\rightarrow \infty} \frac{|\tilde \epsilon^{(t+1)}|}{|\tilde \epsilon^{(t)}|^1} = \frac{1}{2}$$ so the ***bisection method*** behaves like it has ***convergence of order*** $\beta=1$ (i.e., ***linear convergence***), but the convergence of the sequence itself does not have (mathematical) ***convergence of order*** $\beta=1$.
   >
   > Regardless, ***quadratic convergence*** is must faster ***linear convergence***. Recall that for the ***bisection method*** to increase precision by one decimal unit requires about 3.2 iteration steps; whereas, for ***Newton's method*** the precision (approximately) doubles for every single iteration step (assuming the secord order approximation is fairly accurate).

====

Mulitivariate Newton (Return to TOC)

For ***scalar valued multivariate functions*** $g(x)$

> where x is a vector in ${\rm I!R}^d$ but $g(x)$ is scalar valued

the update for ***Newton's Method*** is

$$x^{(t+1)} = x^{(t)} - \left[H_{g(x)}(x^{(t)})\right]^{-1}\nabla_xg(x^{(t)})\quad \text{ in place of } \quad x^{(t+1)} = x^{(t)} - \frac{g'(x^{(t)})}{g''(x^{(t)})}$$

where the effectiveness of this method will be dependent upon the **condition** of $H_{g(x)}(x^{(t)})$, and the method itself is derived from the multivariate generalization of the ***second order Taylor series expansion***

> $$\begin{align*} g(x^) \approx {} & g(x^{(t)}) + (x^ - x^{(t)})^T\nabla_x g(x^{(t)}) + \frac{1}{2}(x^ - x^{(t)})^T H_{g(x)}(x^{(t)})(x^ - x^{(t)})\ \nabla_{x^}g(x^) \approx {} & \nabla_{x^}\left[g(x^{(t)}) + (x^ - x^{(t)})^T\nabla_x g(x^{(t)}) + \frac{1}{2}(x^ - x^{(t)})^T H_{g(x)}(x^{(t)})(x^ - x^{(t)}) \right]\ = {} & \nabla_x g(x^{(t)}) + H_{g(x)}(x^{(t)})(x^ - x^{(t)}) = 0\ \Longrightarrow x^{(t+1)} = {} & x^{(t)} - \left[H_{g(x)}(x^{(t)})\right]^{-1} \nabla_x g(x^{(t)}) \end{align}$$

====

Fisher Scoring (Return to TOC)

For the MLE context where $g(x) \equiv l(\theta)$, the ***expected*** and ***observed*** Fisher information

$$I(\theta) = \overset{\text{expected Fisher information}}{E[\nabla_\theta l(\theta)\nabla_\theta l(\theta)^T]} = E[\overset{\text{observed Fisher}}{\overbrace{-H_{l(\theta)}(\theta)}^{\text{information}}}]$$

suggest a slight variation on ***Newton's method*** known as ***Fisher scoring***

$$\begin{align*} \theta^{(t+1)} & {} = \theta^{(t)} - \left[H_{l(\theta)}(\theta^{(t)})\right]^{-1}\nabla_\theta l(\theta^{(t)}) \ & {} = \theta^{(t)} + \hat{I(\theta^{(t)})}{}^{-1}\nabla_\theta l(\theta^{(t)}) \ & {} \approx \theta^{(t)} + \left[\sum_{i=1}^n \nabla_\theta log f(x_i|\theta)\big|_{\hat \theta}\left(\nabla\theta log f(x_i|\theta)\big|_{\hat \theta}\right)^T \right]^{-1} \left[\sum{i=1}^n \nabla_\theta log f(x_i|\theta)\big|_{\hat \theta}\right] \end{align*}$$

====

Iteratively Reweighted Least Squares (IRLS) (Return to TOC)

In ***exponential family*** form, the ***log likelihood*** of the ***generalized linear model logistic regression*** is

$$l(\beta) = \overbrace{y^t Z\beta - \mathbf{1}^T \underbrace{\log \left(1 + \exp(Z\beta) \right)}_{g(Z\beta)}}^{\log f(y|\theta) := [y\theta - b(\theta)]/a(\phi)+c(y,\phi)}$$

> where
>
> - $\Pr(y_i=1) = \underset{\text{a property of exponential distributions}}{E[y_i] = \frac{d}{d\theta_i} \overbrace{\log \left(1 + \exp(\theta_i) \right)}^{g(\theta_i):=:g(z_i^T\beta)}} = \frac{\exp(\theta_i)}{1 + \exp(\theta_i)} = \frac{1}{1 + \exp(-\theta_i)}$
> - the ***natural parameter*** $\theta_i = z_i^T\beta = \log\left(\frac{\Pr(y_i=1)}{1-\Pr(y_i=1)} \right)$ is the ***log odds***
> - $\log\left(\frac{\Pr(y_i=1)}{1-\Pr(y_i=1)} \right) = \log \left(\frac{\frac{\exp(\theta_i)}{1 + \exp(\theta_i)}}{\frac{1}{1 + \exp(\theta_i)}} \right) = \log(\exp(\theta_i)) = \theta_i$
> - and the ***gradient*** and ***Hessian*** are
>
> $$ \begin{align*} \nabla_\beta l(\beta) & = {}Z^Ty - Z^T\frac{\exp(Z\beta)}{1+\exp(Z\beta)} = Z^Ty - Z^T\frac{1}{1+\exp(-Z\beta)} = Z^T(y-E[y]) \ & = {} Z^Ty - Z^T(1+\exp(-Z\beta))^{-1} \\ H_{l(\beta)}(\beta) & = {} -Z^TWZ \quad \text{ where } \quad W_{ij}=0 \text{ for } i\not=j \ & \quad\quad\quad\quad\quad\quad\quad, \text{ and } \quad W_{ii} = \frac{\exp(-z_i\beta)}

> {\left(1+\exp(-z_i\beta)\right)^2} = \frac{1}{1+\exp(-z_i\beta)}\frac{\exp(-z_i\beta)}{1+\exp(-z_i\beta)} \
> & \quad\quad\quad\quad\quad\quad\quad\quad\quad\quad\quad, = Ey_i \end{align*}$$

Thus, ***Newton's method*** is $$\beta^{(t+1)} = \beta^{(t)} - \underbrace{\overbrace{(-Z^T \overset{(t)}{W}Z)^{-1}}^{H_{l(\beta)}(\beta^{(t)})^{-1}} \overbrace{Z^T(y-\overset{(t)}{E[y]})}^{\nabla_\beta l(\beta^{(t)})}}*{\text{$E[y]$ (and hence W) depends on $\beta^{(t)}$}}$$ which is identical to ***Fisher scoring*** $$\beta^{(t+1)} = \beta^{(t)} + \underbrace{!\hat{I(\beta^{(t)})!}}*{-H_{l(\beta)}(\beta^{(t)})!!{}^{-1} Z^T(y-\overset{(t)}{E[y]})$$

And by reorganizing the terms

> $$\begin{align*} \beta^{(t+1)} = {} & \beta^{(t)} + I(\beta)^{-1} Z^T(y-\overset{(t)}{E[y]})\ = {} & I(\beta)^{-1} \left(I(\beta) \beta^{(t)} + Z^T\underbrace{\overset{(t)}{W}\overset{(t)}{W}{}^{-1}}(y-\overset{(t)}{E[y]})\right)\ = {} & I(\beta)^{-1} \left(\underbrace{Z^T \overset{(t)}{W}Z}*{I(\beta)} \beta^{(t)} + Z^T\overset{(t)}{W} \underbrace{(\tilde y^{(t)}-Z\beta^{(t)})}*{\dagger}\right)\ & \quad\quad;; \dagger \longrightarrow \tilde y^{(t)} = Z\beta^{(t)}+ \overset{(t)}{W}{}^{-1}(y-\overset{(t)}{E[y]}) \ = {} & \underbrace{\left(Z^T \overset{(t)}{W}Z\right)^{-1} !! Z^T\overset{(t)}{W} \tilde y^{(t)}}*{\text{which is weighted least squares}} \end{align*}$$

tbese updates can be recognized as specifying a ***weighted least squares*** problem. Thus, the ***iteratively reweighted least squares*** method fits a ***logistic regression*** by

0. initializing $\beta^{(0)}$ which implies $W^{(t)}$ and $E[y]^{(t)}$

   and then iteratively updating

1. $y^{(t)}$ with $\dagger$
2. $\beta^{(t)}$ with the solution to a ***weighted least squares*** problem.

A presentation of generalized IRLS methods will be covered in the "Gauss-Newton" Section (Section 3.3.0) below.

- The implementation of an IRLS method will be addressed in the second problem of Programming Portfolio Assignment 3.

====

3.1.2 The Secant Method (Return to TOC)

An alternative to ***Newton's method*** for univariate optimization, i.e., finding the roots of $g'(x) = 0$, is the ***secant method***

$$x^{(t+1)} = x^{(t)} - \frac{g'(x^{(t)})(x^{(t)}-x^{(t-1)})}{g'(x^{(t)})-g'(x^{(t-1)})} \quad \text{ which replaces } \quad x^{(t+1)} = x^{(t)} - \frac{g'(x^{(t)})}{g''(x^{(t)})}$$

by specifing a sequence of two initial values $x^{(-1)}$ and $x^{(0)}$ and then substituting the ***discrete difference*** approximation of the second derivative g'' at $x^{(t)}$

$$g''(x^{(t)}) \approx \frac{g'(x^{(t)})-g'(x^{(t-1)})}{x^{(t)}-x^{(t-1)}} $$

> The ***secant method*** can be shown to have a convergence order $\beta = (1+\sqrt{5})/2 \approx 1.62$ (i.e., better than ***linear*** but worse than ***quadratic convergence***), so approximating the derivative causes the ***order of convergence*** $\beta \approx 1.62$ of the ***secant methods*** to be worse than the ***order of convergence*** $\beta = 2$ of ***Newton's Method***. However
>
> - ***Newton's Method*** requires two function evaluations per iteration, i.e., $g'(x^{(t)})$ and $g''(x^{(t)})$
> - but the ***secant methods*** only requires a single function evaluation per iteration, i.e., $g'(x^{(t)})$ may be used at step t and reused on step $t-1$
>
> so the actual computational time requirements of the two methods are actually more competitive than is suggested by the theoretical ***order of convergence***. This is precisely the "per step cost versus number of steps" tradeoff endemic to ***iterative methods*** noted in the introduction (Section 3.1).