$Y_1, \dots, Y_n$ iid

$$f(y;\lambda) = \frac{\lambda^{y_i} e^{-\lambda}}{y_i!\,(1-e^{-\lambda})} \qquad y_i = 1, 2, \dots$$

$$\ell(\lambda) = (\Sigma y_i)\cdot \log\lambda - n\lambda - n\log(1-e^{-\lambda})$$

$$= n\left(\bar{y}\log\lambda - \lambda - \log(1-e^{-\lambda})\right)$$

$$\ell'(\lambda) = n\left(\frac{\bar{y}}{\lambda} - 1 - \frac{e^{-\lambda}}{1-e^{-\lambda}}\right)$$

$$= n\left(\frac{\bar{y}}{\lambda} - \frac{(1-e^{-\lambda})}{(1-e^{-\lambda})} - \frac{e^{-\lambda}}{1-e^{-\lambda}}\right)$$

$$= n\left(\frac{\bar{y}}{\lambda} - \frac{1}{1-e^{-\lambda}}\right)$$

$$\ell'(\hat{\lambda}) = 0 \qquad \frac{\bar{y}}{\hat{\lambda}} = \frac{1}{1-e^{-\hat{\lambda}}}$$

$$\boxed{\hat{\lambda} = \bar{y}\,(1-e^{-\hat{\lambda}})}$$

$$\hat{\lambda}^{(0)} = \bar{y}$$

$$\hat{\lambda}^{(t+1)} = \bar{y}\,(1-e^{-\hat{\lambda}^{(t)}})$$ simple iteration

(NR) & $$\hat{\lambda}^{(t+1)} = \hat{\lambda}^{(t)} - \frac{\ell'(\hat{\lambda}^{(t)})}{\ell''(\hat{\lambda}^{(t)})}$$

Solve $f(x) = 0$

Iteration: find $g(x)$ s.t. $g(x) = x$

$\boxed{g(x) = f(x) + x} \Rightarrow f(x) = 0$

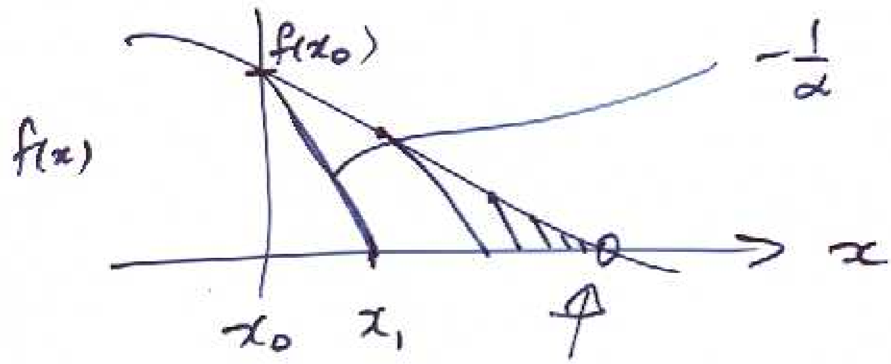
$$x^{(t+1)} = g(x^{(t)}) = \underline{x^{(t)}} + \underline{f(x^{(t)})}$$

$g(x) = x$
$\Rightarrow f(x) = 0$
then
$g(x) = x + f(x)$

$$x^{(t+1)} = x^{(t)} + \alpha f(x^{(t)})$$

for some choice α $\qquad \alpha \in \dfrac{1}{f'(x^{(t)})}$

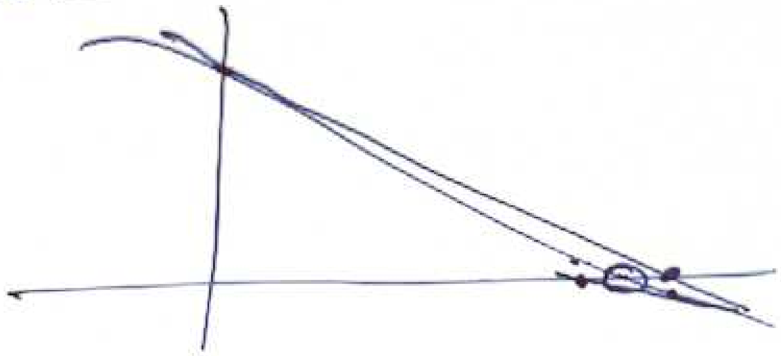
NR



↑ Simple iter.

If we replace α by $\dfrac{1}{f'(x^{(t)})}$, then

we get NR



nzp. ~~newton~~ method.of. scoring $\leftarrow$ special to log-lik.

$$l'(\hat\theta) = 0 = l'(\theta) + (\hat\theta - \theta)\, l''(\theta)$$

$$\hat\theta \simeq \theta - \frac{l'(\theta)}{l''(\theta)}$$

replace $l''(\theta)$ by $\underbrace{E(l''(\theta))}_{=-i(\theta)}$ $\leftarrow$ "scoring"

$$\hat\theta^{(t+1)} = \hat\theta^{(t)} + \frac{l'(\hat\theta^{(t)})}{i(\hat\theta^{(t)})}$$

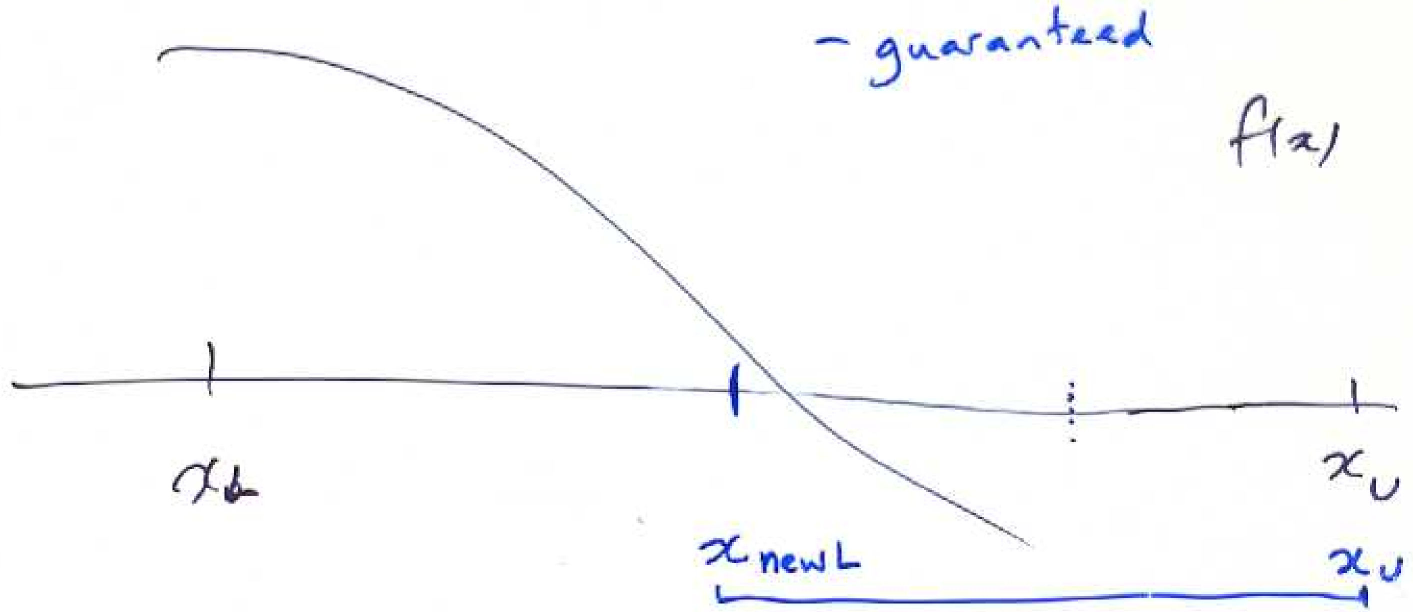
replace obs'd inf. by exp'd

In R, solving $f(x) = 0$, most often use <u>uniroot</u>

Brent's method
- bisection
- clever adjustm.
- interpolation
- better convergence than bisection
- guaranteed

(doesn't use derivatives)

- bracketing method



input an interval that brackets the root

bisection   cut interval in half

- guaranteed to find a root if there is one
- slow