HW2 Stat-comp (due Wed, Oct 19th in D2L)

1) Maximum likelihood estimation and inference with the exponential distribution

The density function of an exponential random variable is

$f(x_i|\lambda) = \lambda e^{-\lambda x_i}$

where $x_i \geq 0$ is the random variable, and $\lambda > 0$ is a rate parameter.

The expected value and variance of the random variables are $E[X] = \frac{1}{\lambda}$ and $Var[X] = \frac{1}{\lambda^2}$.

The following code simulates 50 IID draws from an exponential distribution

```
set.seed(195021)
x=rexp(n=50,rate=2)
```

The maximum likelihood estimate of λ has a closed form, indeed

$L(\lambda|x) = \lambda^n e^{-\lambda n\bar{x}}$

Thus, $l(\lambda|x) = nlog(\lambda) - \lambda n\bar{x}$, therefore

$\frac{dl}{d\lambda} = \frac{n}{\lambda} - n\bar{x}$. Setting this derivative equal to zero, and solving for $\hat{\lambda}$ gives $\hat{\lambda} = \frac{1}{\bar{x}}$

Using numerical optimization to estimate λ:

Since $\lambda > 0$, we need to be careful using `optim()` because this function may report an estimate smaller than zero. Furthermore, for models involving a single parameter, `optimize()` is preferred relative to `optim()`; `optimize()` allows you to provide an interval for the optimization.

1.1) Use `optimize()` to estimate λ compare your estimate with $\frac{1}{\bar{x}}$.

1.2) Use numerical methods to proivde an approximate 95% CI for your estimate.

Hint: `optimize()` does not provide a Hessian. However, you can use the `hessian()` function of the `numDeriv` R-package to obtain a numerical approximation to the second order derivative of the logLikelihood at the ML estiamte. To install this package you can use

```
install.packages(pkg='numDeriv',repos='https://cran.r-project.org/')
```

2) CIs for Predictions from Logistic Regression

Recall that in a logistic regresion model, the log-odds are parameterized as

$$log[\frac{\theta_i}{(1-\theta_i)}] = \mathbf{x}_i'\beta = \eta_i \tag{1}$$

The sampling variance of $\mathbf{x}_i'\beta = \eta_i$ is $Var(\eta_i) = \mathbf{x}_i'\mathbf{V}\mathbf{x}_i$, where $\mathbf{V}$ is the (co)variance matrix of the estimated effects; therefore, a SE and an approximate 95%CI for η_i can be obtained using

$SE(\eta_i) = \sqrt{\mathbf{x}_i'\mathbf{V}\mathbf{x}_i}$ and $CI: \mathbf{x}_i'\hat{\beta} +/- 1.96 \times SE(\eta_i)$.

Because the inverse-logit is a monotonic map, we can then obtain a 95% CI for the predicted probabilities by applying the inverse logit, $\theta_i = \frac{e^{\eta_i}}{1+e^{\eta_i}}$, to the bounds of the CI for the linear predictor.

- Using the gout data set, fit a logistic regression for gout using sex, age, and race as predictors (for this you can use `glm()`, don't forget the link!).
- From the fitted model, and using the formulas presented above, compute the predicted probability of gout for each of the following cases, and the corresponding 95% CI for the predicted risk.

Race	Sex	Age	Predicted Risk	95%CI
White	Male	55		
White	Female	55		
Black	Male	55		
Black	Female	55		

3) Bootstrap/

Use 1,000 bootstrap samples to estimate the SE and 95% CIs for the probabilities reported in Question 2. Compare your bootsrap results with those reported in Question 2.