Curve_fitting_HSP_performance_Curves

Andrew Nguyen

2015-May-11

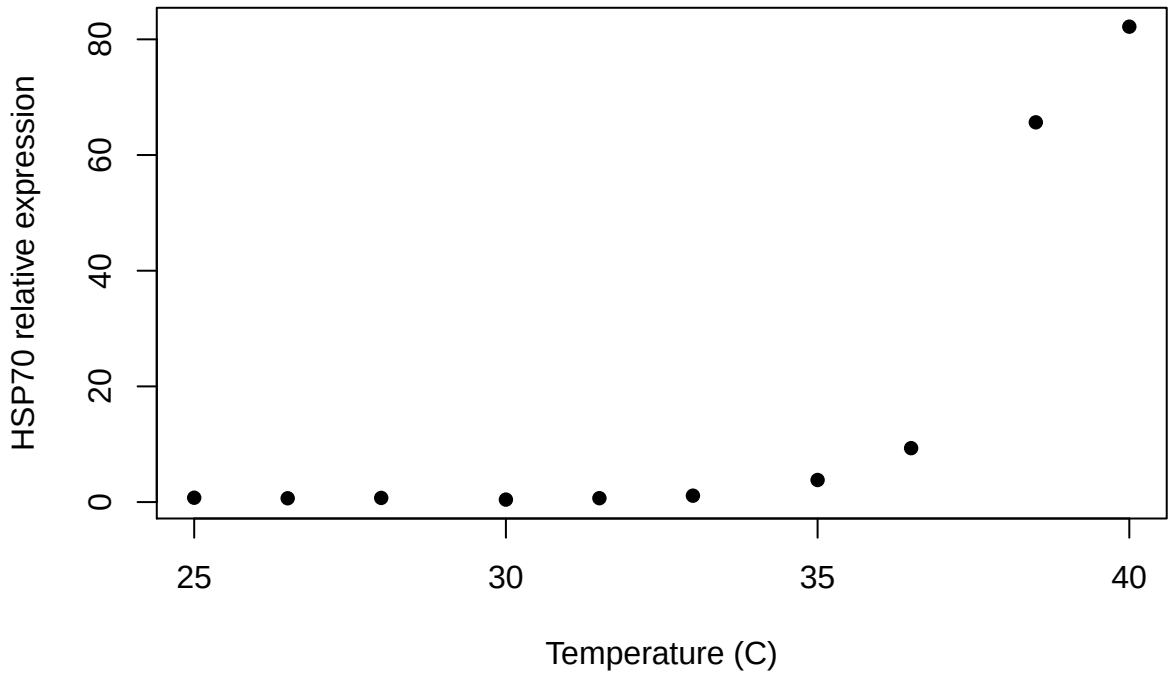
Trying to fit curves: From Sarah Diamond– Obviously 3 points is really the minimum for fitting a curve (be prepared for some criticism here regarding index temperatures when this work is reviewed). This becomes much easier if you have multiple replicates, say reaction norms for multiple individuals. While Im guessing its not possible to get greater numbers of index temperatures, having the extra replicates will make the curve fitting less arduous. Because its unclear what the expectation should be for these curves, my suggestion would be to fit multiple curves, and use AIC to choose the best-fitting model. Non-linear least squares (nls function from nlme in R) with Gaussian, modified Gaussian, beta, Weibull, (each of these have different parameter variants my guess is that the fewer parameter versions would be preferable owing to limited index) would be a good place to start.

```
library(nlme)
library(numDeriv)
```

```
y<-read.csv("2015_ANBE_common_garden_gxp_evolution.csv")
head(y)
```

```
##   N     species colony temp FC_hsc70.4h2  FC_hsp83  FC_hsp40
## 1 1 lamellidens  duke8 25.0    0.7481776 0.7646694 0.8134659
## 2 2 lamellidens  duke8 26.5    0.6579517 0.5729690 0.5776902
## 3 3 lamellidens  duke8 28.0    0.7169249 1.1590576 0.7289012
## 4 4 lamellidens  duke8 30.0    0.4289565 1.1042502 0.7912562
## 5 5 lamellidens  duke8 31.5    0.6719769 0.7786837 0.6099290
## 6 6 lamellidens  duke8 33.0    1.1053425 0.6237785 0.9379909
```

```
x<-y[1:10,]
x$temp<-as.numeric(as.character(x$temp))

plot(x$temp,x$FC_hsc70.4h2,xlab="Temperature (C)",ylab="HSP70 relative expression",pch=16,col="black")
```



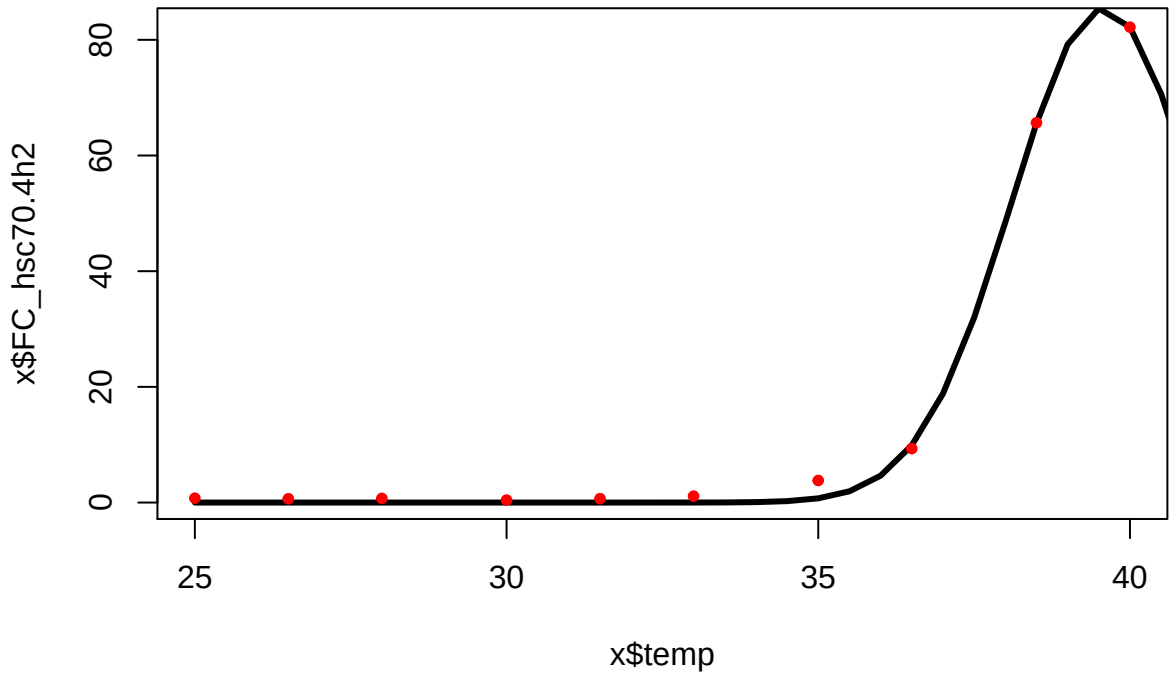

```
#sarah diamond's sample function
#Gaussian curve
G<-nls(FC_hsc70.4h2 ~ a*exp(-0.5*((temp-b)/c)^2),data=x, start=list(a=25, b=40, c=1.5), trace=TRUE,cont
```

```
## 5895.902 :  25.0 40.0  1.5
## 1755.842 :  82.200136 38.557874  1.615628
## 435.8948 :  66.998164 39.557755  1.873479
## 36.79777 :  83.740795 39.749138  1.558776
## 13.98664 :  85.021042 39.561643  1.479833
## 13.23483 :  85.535218 39.582109  1.483813
## 13.23476 :  85.542296 39.582086  1.483757
## 13.23476 :  85.542362 39.582083  1.483752
```

```
summary(G)
```

```
##
## Formula: FC_hsc70.4h2 ~ a * exp(-0.5 * ((temp - b)/c)^2)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a 85.54236    1.40940   60.69 8.65e-11 ***
## b 39.58208    0.05774  685.57  < 2e-16 ***
## c  1.48375    0.06844   21.68 1.12e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.375 on 7 degrees of freedom
##
## Number of iterations to convergence: 7
## Achieved convergence tolerance: 2.629e-06
```

```r
plot(x$temp,x$FC_hsc70.4h2,type='n')
new<-seq(25,41,.5)
lines(new,predict(G,list(temp=new)),lwd=3)
points(x$temp,x$FC_hsc70.4h2,pch=20,col="red")
```




```r
#lines(G, predict(x, list(x = G)), col = "green")
#modified Gaussian
#beta

#Weibull
#1-e(-(x/lambda)^k)

#exp(-(temp/a)^b)
#W<-nls(FC_hsc70.4h2 ~ 1-exp(-(temp/lam)^k),data=x, start=list(lam=.5,k=1.5), trace=TRUE,control=nls.co

#nls(hsc4.2 ~ c*((Temp/a)^(b-1))*exp(-(Temp/a)^b), start = list(a = 30, b = 30,c = 1), data = x,trace=T
```

Boltzmann fit

```r
#Boltzman
#y = LL + (UL- LL)/ (1+exp(Tm-x/a))
#c= max expression
#b= expression value at inflection point
B<-nls(FC_hsc70.4h2 ~ (1+(max-1)/(1+exp((inflect-temp)/a))),data=x, start=list(max=80,inflect=35,a=1.05)
```

```
## 4672.776 :  80.00 35.00  1.05
## 1438.282 :  83.627101 37.568955  2.123639
## 909.2676 :  112.519266  39.366313   1.832872
```

```
## 761.2292 :  75.437510 37.804349  1.251989
## 152.1986 :  77.9670212 37.6497135  0.7854722
## 5.507782 :  84.0801825 37.7607431  0.5752434
## 5.109379 :  84.045466 37.764423  0.587678
## 5.108606 :  84.0612501 37.7644931  0.5884446
## 5.108604 :  84.0619582 37.7644852  0.5884851
## 5.108604 :  84.0619983 37.7644849  0.5884874
```

```
summary(B)
```

```
##
## Formula: FC_hsc70.4h2 ~ (1 + (max - 1)/(1 + exp((inflect - temp)/a)))
##
## Parameters:
##         Estimate Std. Error t value Pr(>|t|)
## max     84.06200    1.06880   78.65 1.41e-11 ***
## inflect 37.76448    0.04115  917.69  < 2e-16 ***
## a        0.58849    0.02424   24.28 5.12e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8543 on 7 degrees of freedom
##
## Number of iterations to convergence: 9
## Achieved convergence tolerance: 1.922e-06
```

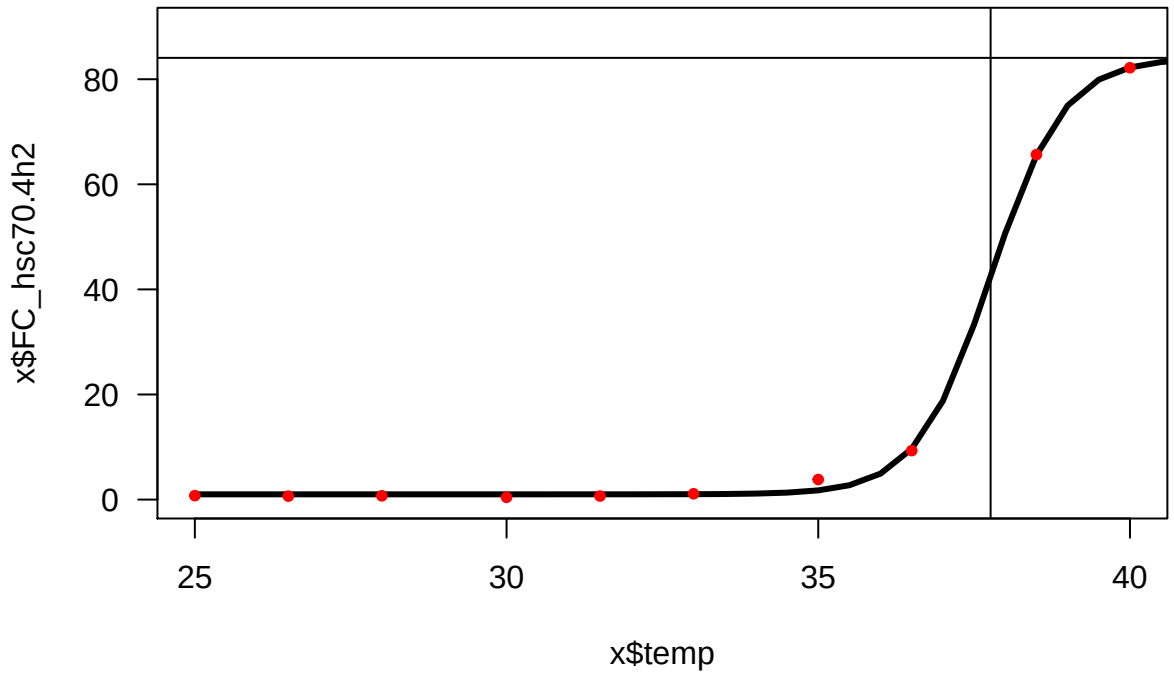
```
summary(B)$parameters
```

```
##           Estimate Std. Error   t value     Pr(>|t|)
## max     84.0619983 1.06879870  78.65092 1.413581e-11
## inflect 37.7644849 0.04115161 917.69158 4.818221e-19
## a        0.5884874 0.02424110  24.27643 5.123021e-08
```

```
#expression value at inflection point
summary(B)$parameters[2,1]
```

```
## [1] 37.76448
```

```
plot(x$temp,x$FC_hsc70.4h2,type='n',las=1,ylim=c(0,90))
#lines(x$Temp,predict(B),lwd=3)
lines(new,predict(B,list(temp=new)),lwd=3)
points(x$temp,x$FC_hsc70.4h2,pch=20,col="red")
#plotting inflection point
abline(v=summary(B)$parameters[2,1])
#max point
abline(h=summary(B)$parameters[1,1])
```



##AIC

```
AIC(G,B)
```

```
##   df      AIC
## G  4 39.18139
## B  4 29.66218
```

ted4; picea

```
x<-y[11:20,]
x$temp<-as.numeric(as.character(x$temp))

G<-nls(FC_hsc70.4h2 ~ a*exp(-0.5*((temp-b)/c)^2),data=x, start=list(a=25, b=40, c=1.5), trace=TRUE,cont
```

```
## 6108.913 :  25.0 40.0  1.5
## 759.1337 :  67.44409 38.61214  2.57338
## 373.2762 :  60.752308 39.330509  2.760384
## 335.12 :  64.447345 39.333545  2.549603
## 333.2439 :  65.18023 39.31866  2.49098
## 333.0639 :  65.344490 39.318625  2.475157
## 333.0435 :  65.398060 39.317959  2.469478
## 333.0411 :  65.415682 39.317817  2.467591
## 333.0408 :  65.421732 39.317759  2.466939
## 333.0408 :  65.423798 39.317741  2.466716
## 333.0408 :  65.42451 39.31773  2.46664
## 333.0408 :  65.424751 39.317732  2.466613
## 333.0408 :  65.424835 39.317731  2.466604
```

```r
summary(G)
```

```
##
## Formula: FC_hsc70.4h2 ~ a * exp(-0.5 * ((temp - b)/c)^2)
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## a  65.4248     5.2995  12.345 5.25e-06 ***
## b  39.3177     0.3204 122.696 6.30e-13 ***
## c   2.4666     0.3577   6.895 0.000232 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.898 on 7 degrees of freedom
##
## Number of iterations to convergence: 12
## Achieved convergence tolerance: 3.773e-06
```

```r
B<-nls(FC_hsc70.4h2 ~ (1+(max-1)/(1+exp((inflect-temp)/a))),data=x, start=list(max=80,inflect=35,a=1.05)
```

```
## 3357.968 :  80.00 35.00  1.05
## 654.6082 :  60.869630 35.696277  1.422011
## 597.6435 :  61.335919 36.141724  1.261025
## 596.9875 :  60.793372 36.108761  1.167654
## 596.8584 :  61.587605 36.180890  1.236857
## 596.3528 :  61.212790 36.149392  1.201885
## 596.3499 :  61.228341 36.151359  1.200314
## 596.3498 :  61.233709 36.152136  1.200481
## 596.3498 :  61.229916 36.151872  1.200074
## 596.3498 :  61.233838 36.152217  1.200422
## 596.3498 :  61.230093 36.151900  1.200077
## 596.3498 :  61.233722 36.152209  1.200409
## 596.3498 :  61.230214 36.151911  1.200088
## 596.3498 :  61.233606 36.152199  1.200398
## 596.3498 :  61.230327 36.151920  1.200098
## 596.3498 :  61.233498 36.152190  1.200388
## 596.3498 :  61.230432 36.151929  1.200108
## 596.3498 :  61.233397 36.152181  1.200379
## 596.3498 :  61.230529 36.151938  1.200117
## 596.3498 :  61.233302 36.152173  1.200371
## 596.3498 :  61.230621 36.151945  1.200125
## 596.3498 :  61.233213 36.152166  1.200362
## 596.3498 :  61.230707 36.151953  1.200133
## 596.3498 :  61.233132 36.152159  1.200355
## 596.3498 :  61.23079 36.15196  1.20014
## 596.3498 :  61.233053 36.152152  1.200348
## 596.3498 :  61.230863 36.151966  1.200147
## 596.3498 :  61.232981 36.152146  1.200341
## 596.3498 :  61.230932 36.151972  1.200154
## 596.3498 :  61.232913 36.152140  1.200335
## 596.3498 :  61.23100 36.15198  1.20016
## 596.3498 :  61.232850 36.152135  1.200329
```

```
## 596.3498 :  61.231059 36.151983  1.200165
## 596.3498 :  61.232790 36.152130  1.200324
## 596.3498 :  61.23112 36.15199  1.20017
## 596.3498 :  61.232735 36.152125  1.200319
## 596.3498 :  61.231170 36.151992  1.200175
## 596.3498 :  61.232683 36.152121  1.200314
## 596.3498 :  61.23122 36.15200  1.20018
## 596.3498 :  61.232635 36.152117  1.200309
## 596.3498 :  61.231267 36.152000  1.200184
## 596.3498 :  61.232591 36.152113  1.200305
## 596.3498 :  61.231309 36.152004  1.200188
## 596.3498 :  61.232549 36.152109  1.200302
## 596.3498 :  61.231350 36.152007  1.200192
## 596.3498 :  61.232509 36.152106  1.200298
## 596.3498 :  61.231389 36.152011  1.200195
## 596.3498 :  61.232472 36.152103  1.200294
## 596.3498 :  61.231425 36.152014  1.200199
## 596.3498 :  61.232437 36.152100  1.200291
## 596.3498 :  61.231459 36.152017  1.200202
## 596.3498 :  61.232404 36.152097  1.200288
## 596.3498 :  61.231490 36.152019  1.200205
## 596.3498 :  61.232374 36.152094  1.200286
## 596.3498 :  61.231519 36.152022  1.200207
## 596.3498 :  61.232346 36.152092  1.200283
## 596.3498 :  61.23155 36.15202  1.20021
## 596.3498 :  61.232320 36.152090  1.200281
## 596.3498 :  61.231572 36.152026  1.200212
## 596.3498 :  61.232295 36.152088  1.200278
## 596.3498 :  61.231596 36.152028  1.200214
## 596.3498 :  61.232271 36.152086  1.200276
## 596.3498 :  61.231619 36.152030  1.200216
## 596.3498 :  61.232250 36.152084  1.200274
## 596.3498 :  61.231639 36.152032  1.200218
## 596.3498 :  61.232230 36.152082  1.200272
## 596.3498 :  61.23166 36.15203  1.20022
## 596.3498 :  61.232211 36.152081  1.200271
## 596.3498 :  61.231676 36.152035  1.200222
## 596.3498 :  61.232193 36.152079  1.200269
## 596.3498 :  61.231694 36.152037  1.200223
## 596.3498 :  61.232177 36.152078  1.200267
## 596.3498 :  61.231710 36.152038  1.200225
## 596.3498 :  61.232161 36.152076  1.200266
## 596.3498 :  61.231725 36.152039  1.200226
## 596.3498 :  61.232146 36.152075  1.200265
## 596.3498 :  61.231739 36.152040  1.200227
## 596.3498 :  61.232134 36.152074  1.200264
## 596.3498 :  61.231752 36.152042  1.200229
## 596.3498 :  61.232121 36.152073  1.200262
## 596.3498 :  61.23176 36.15204  1.20023
## 596.3498 :  61.232108 36.152072  1.200261
## 596.3498 :  61.231777 36.152044  1.200231
## 596.3498 :  61.23210 36.15207  1.20026
## 596.3498 :  61.231787 36.152045  1.200232
## 596.3498 :  61.232086 36.152070  1.200259
```

```
## 596.3498 :  61.231798 36.152045  1.200233
## 596.3498 :  61.232076 36.152069  1.200258
## 596.3498 :  61.231807 36.152046  1.200234
## 596.3498 :  61.232067 36.152068  1.200257
## 596.3498 :  61.231816 36.152047  1.200234
## 596.3498 :  61.232059 36.152068  1.200257
## 596.3498 :  61.231824 36.152048  1.200235
## 596.3498 :  61.232051 36.152067  1.200256
## 596.3498 :  61.231832 36.152048  1.200236
## 596.3498 :  61.232043 36.152066  1.200255
## 596.3498 :  61.231839 36.152049  1.200237
## 596.3498 :  61.232036 36.152066  1.200255
## 596.3498 :  61.231845 36.152050  1.200237
## 596.3498 :  61.232030 36.152065  1.200254
```

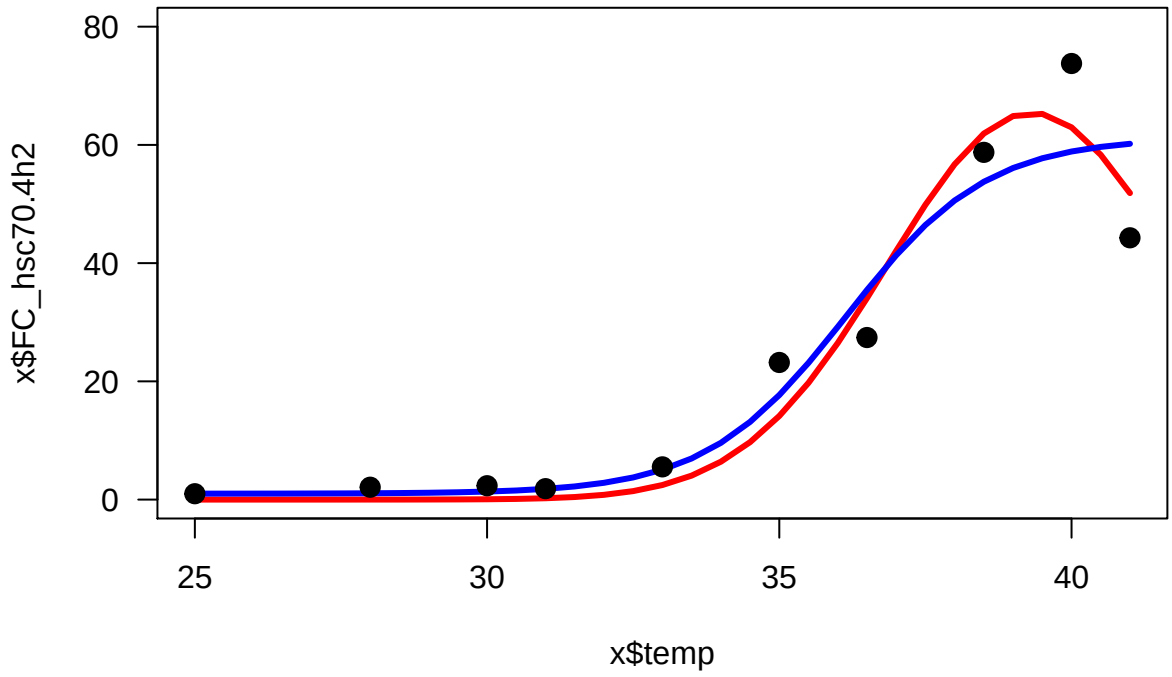
```
summary(B)
```

```
##
## Formula: FC_hsc70.4h2 ~ (1 + (max - 1)/(1 + exp((inflect - temp)/a)))
##
## Parameters:
##         Estimate Std. Error t value Pr(>|t|)
## max      61.2320     8.7172   7.024 0.000207 ***
## inflect  36.1521     0.7524  48.050 4.42e-10 ***
## a         1.2003     0.6686   1.795 0.115692
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.23 on 7 degrees of freedom
##
## Number of iterations to convergence: 99
## Achieved convergence tolerance: 9.939e-06
```

```
summary(B)$parameters
```

```
##          Estimate Std. Error   t value     Pr(>|t|)
## max     61.232030  8.7172366  7.024248 2.070514e-04
## inflect 36.152065  0.7523866 48.049851 4.424258e-10
## a        1.200254  0.6685849  1.795216 1.156916e-01
```

```
plot(x$temp,x$FC_hsc70.4h2,type='n',las=1,ylim=c(0,80))
#lines(x$Temp,predict(B),lwd=3)
lines(new,predict(G,list(temp=new)),lwd=3,col="red")
lines(new,predict(B,list(temp=new)),lwd=3,col="blue")
points(x$temp,x$FC_hsc70.4h2,pch=20,col="black",cex=2)
```




```
AIC(G,B)
```

```
##   df      AIC
## G  4 71.43557
## B  4 77.26119
```

took the derivative from http://www.derivative-calculator.net/

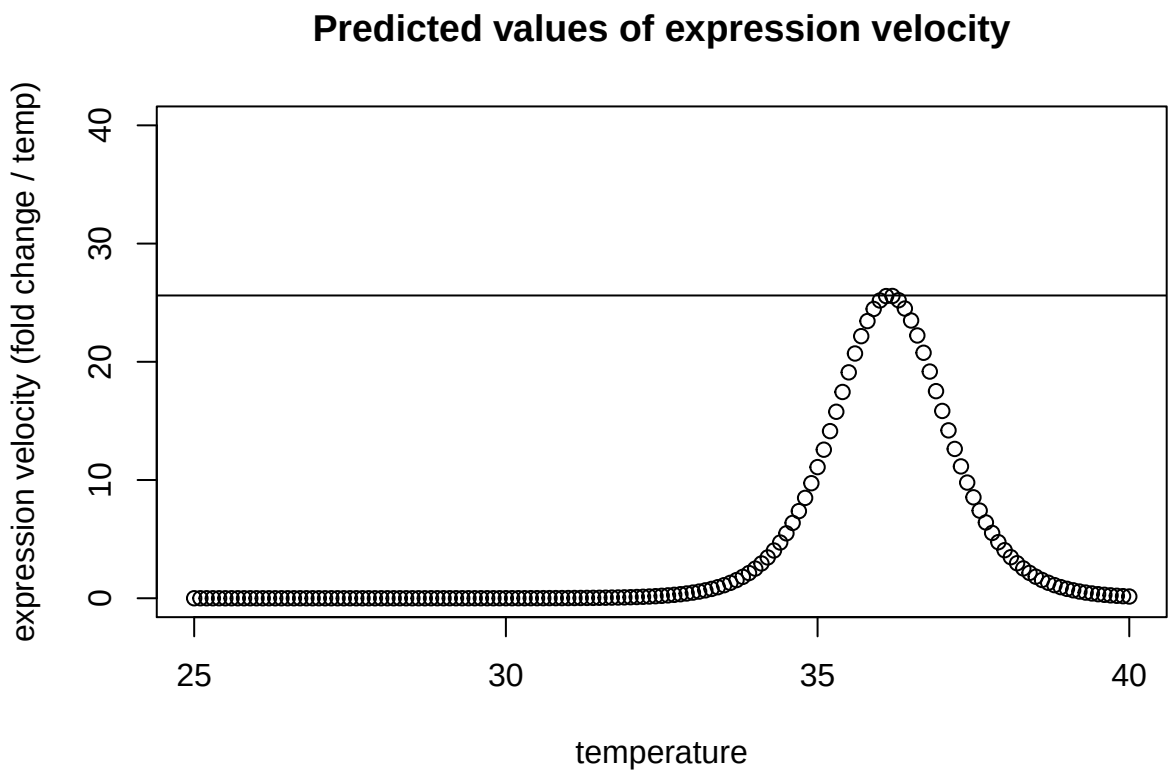
((min-max)*eˆ((T-x)/a)) / (a(e$^{((\text{inflect-x})/a)+1)}$2)

Create new function and plug in inflection point to grab instantaneous velocity

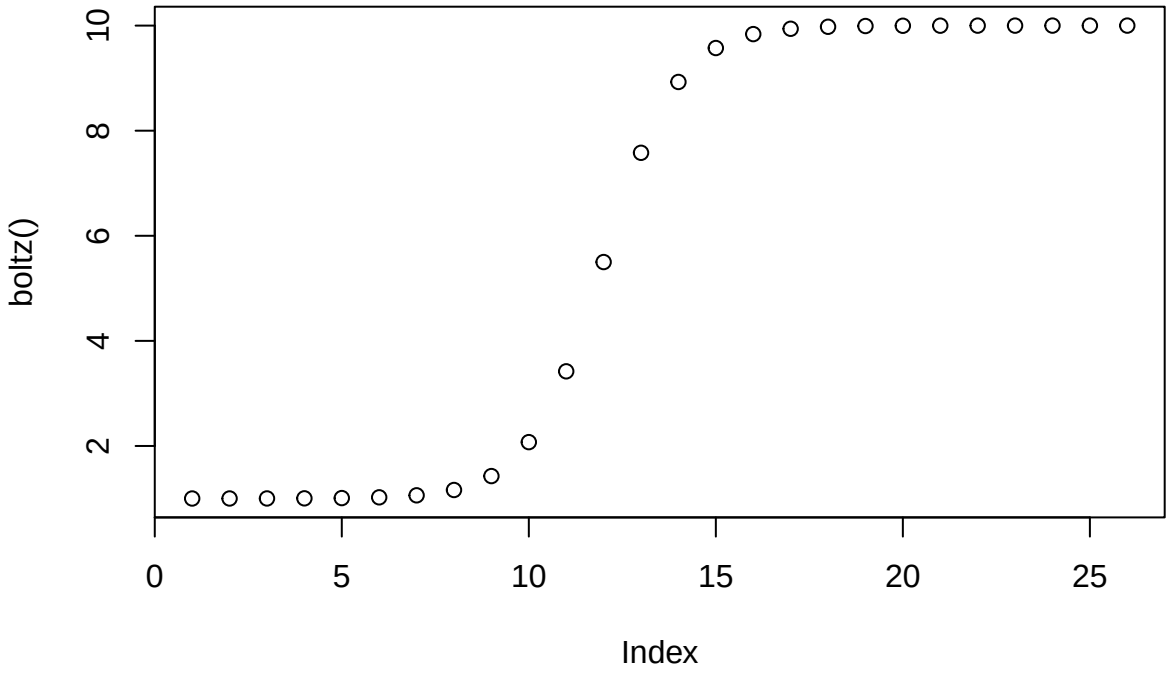
```
#derivative function!
maxi<-function(max=80,inflect=36,a=.588,temp=seq(25,50,1)){
  y= ((max-1)*exp((inflect-temp)/a))/ (a*(exp((inflect-temp)/a)+1)^2)
return(y)
}

#finding slope at inflection point
slope<-maxi(max=summary(B)$parameters[1,1],inflect=summary(B)$parameters[2,1],temp=summary(B)$parameters
slope
```

```
## [1] 25.60886
```

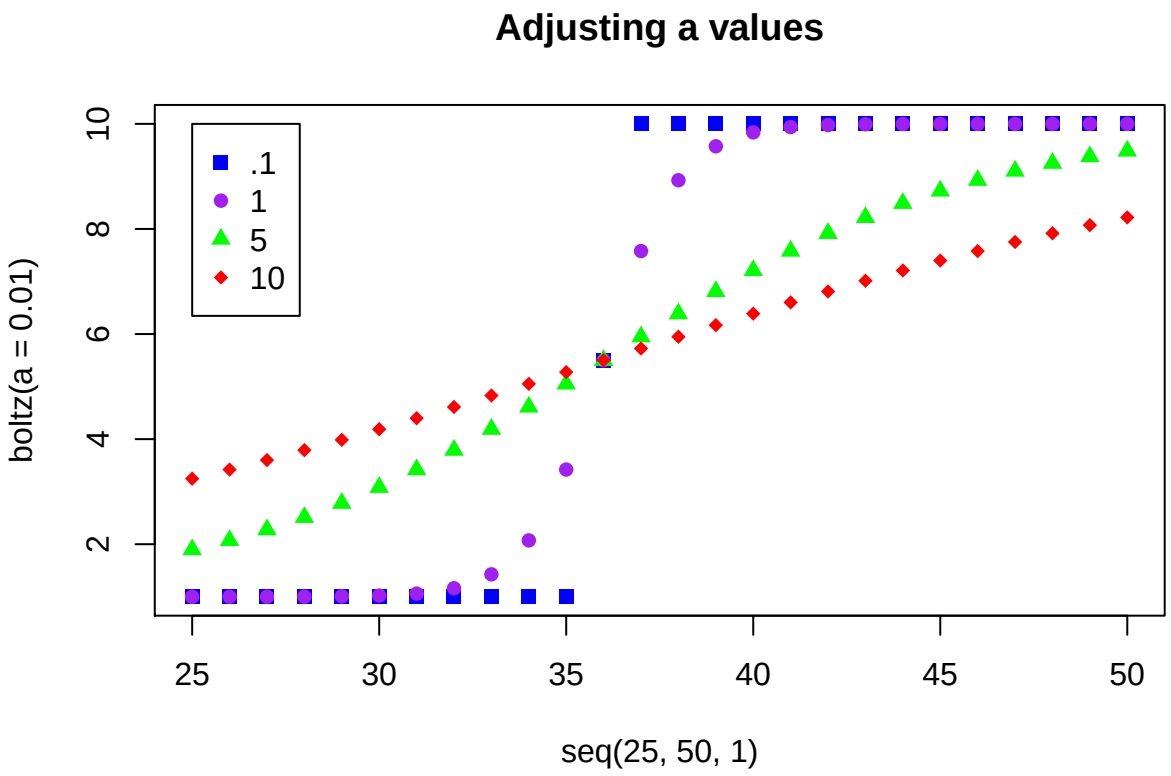
```
plot(seq(25,40,0.1),maxi(max=summary(B)$parameters[1,1],inflect=summary(B)$parameters[2,1],temp=seq(25,4
abline(h=slope)
```



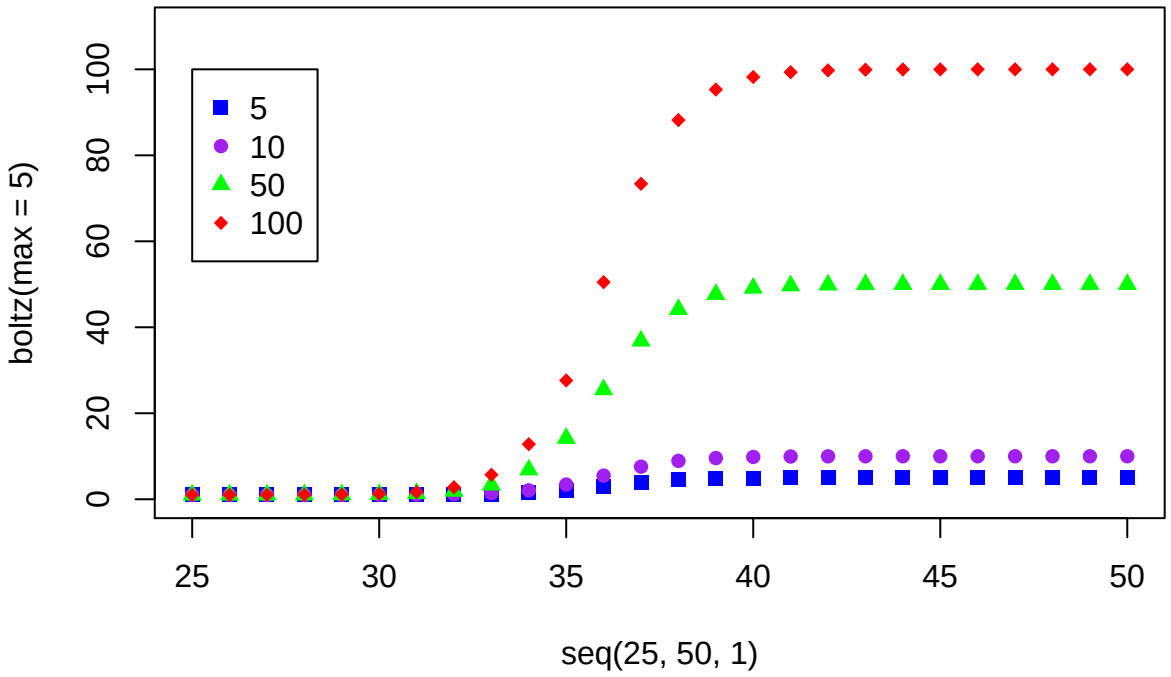

```
boltz<-function(max=10,min=1,a=1,Tm=36,temp=seq(25,50,1)){
  y=min+((max-min)/(1+exp((Tm-temp)/a)))
return(y)                                                    }

#plotting boltz function
plot(boltz())
```



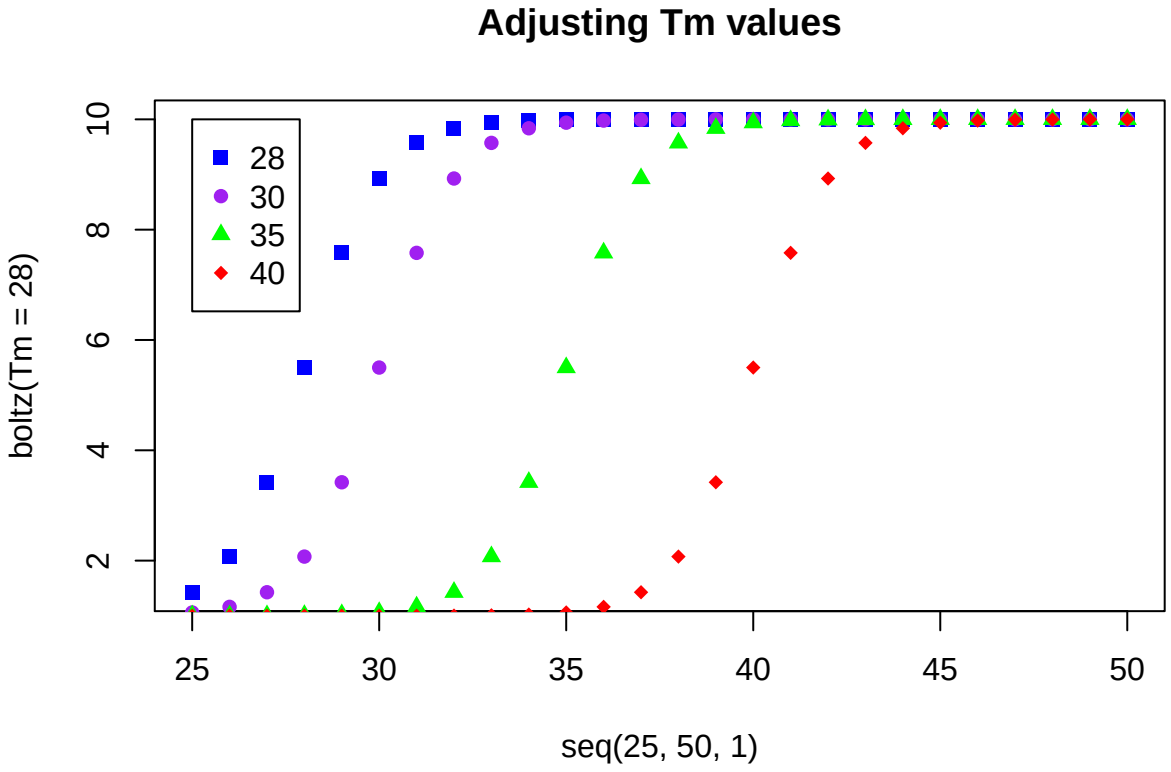
```r
#adjusting a
plot(seq(25,50,1),boltz(a=.01),pch=15,col="blue",main="Adjusting a values")
points(seq(25,50,1),boltz(a=1),pch=16,col="purple")
points(seq(25,50,1),boltz(a=5),pch=17,col="green")
points(seq(25,50,1),boltz(a=10),pch=18,col="red")

legend(25,10,c(".1","1","5","10"),pch=seq(15,18,1),col=c("blue","purple","green","red"))
```




```r
#ok let's change max values
plot(seq(25,50,1),boltz(max=5),pch=15,col="blue",main="Adjusting max values",ylim=c(0,110))
points(seq(25,50,1),boltz(max=10),pch=16,col="purple")
points(seq(25,50,1),boltz(max=50),pch=17,col="green")
points(seq(25,50,1),boltz(max=100),pch=18,col="red")

legend(25,100,c("5","10","50","100"),pch=seq(15,18,1),col=c("blue","purple","green","red"))
```




```
#A little silly...

#Lets change TM
plot(seq(25,50,1),boltz(Tm=28),pch=15,col="blue",main="Adjusting Tm values")
points(seq(25,50,1),boltz(Tm=30),pch=16,col="purple")
points(seq(25,50,1),boltz(Tm=35),pch=17,col="green")
points(seq(25,50,1),boltz(Tm=40),pch=18,col="red")

legend(25,10,c("28","30","35","40"),pch=seq(15,18,1),col=c("blue","purple","green","red"))
```




```
sessionInfo()
```

```
## R version 3.2.0 (2015-04-16)
## Platform: x86_64-apple-darwin13.4.0 (64-bit)
## Running under: OS X 10.10.2 (Yosemite)
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] numDeriv_2014.2-1 nlme_3.1-120
##
## loaded via a namespace (and not attached):
##  [1] magrittr_1.5    formatR_1.2     tools_3.2.0     htmltools_0.2.6
##  [5] yaml_2.1.13     stringi_0.4-1   rmarkdown_0.6.1 grid_3.2.0
##  [9] knitr_1.10.5    stringr_1.0.0   digest_0.6.8    lattice_0.20-31
## [13] evaluate_0.7
```