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.

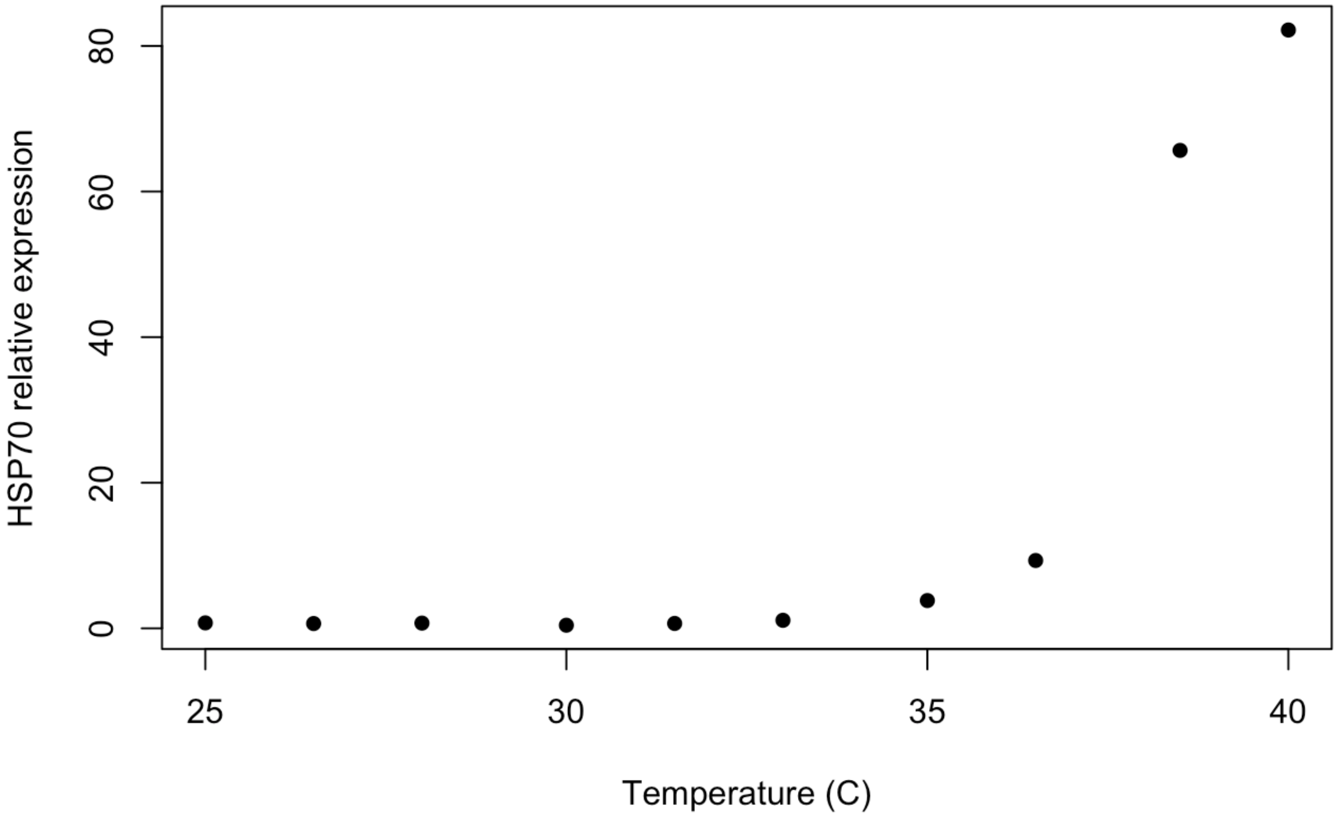
```
library(nlme)
```

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

```
##   N     species colony temp gapdh_CT  ef1b_CT actin_CT hsc70.4h2_CT
## 1 1 lamellidens  duke8   25 21.60177 26.15980 21.54313     21.83867
## 2 2 lamellidens  duke8 26.5 20.91830 25.79680 21.48074     21.34059
## 3 3 lamellidens  duke8   28 21.22636 25.57809 21.04811     21.52481
## 4 4 lamellidens  duke8   30 21.15318 25.89326 21.57288     22.19263
## 5 5 lamellidens  duke8 31.5 20.96067 25.79220 21.76471     21.35253
## 6 6 lamellidens  duke8   33 21.08613 26.15698 21.44401     20.75999
##   hsp83_CT hsp40_CT  geomean deltct_HKG delta.hsc704 delta_hsp83
## 1 23.99627 29.25980 23.00440  0.2137925   -0.2047548 -0.17329947
## 2 23.72918 29.07012 22.63153  0.8972632    0.2933168  0.09379228
## 3 23.02082 29.04275 22.52435  0.5892026    0.1090965  0.80215486
## 4 23.01753 28.85115 22.77667  0.6623789   -0.5587177  0.80544599
## 5 23.32897 29.03414 22.74480  0.8548953    0.2813788  0.49400457
## 6 23.77444 28.53867 22.78401  0.7294308    0.8739243  0.04853662
##   delta_HSP40 FC_hsc70.4h2  FC_hsp83  FC_hsp40
## 1 -0.08405367    0.7481776 0.7646694 0.8134659
## 2  0.10563119    0.6579517 0.5729690 0.5776902
## 3  0.13299783    0.7169249 1.1590576 0.7289012
## 4  0.32459577    0.4289565 1.1042502 0.7912562
## 5  0.14160856    0.6719769 0.7786837 0.6099290
## 6  0.63707670    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 expressio
n",pch=16,col="black")
```



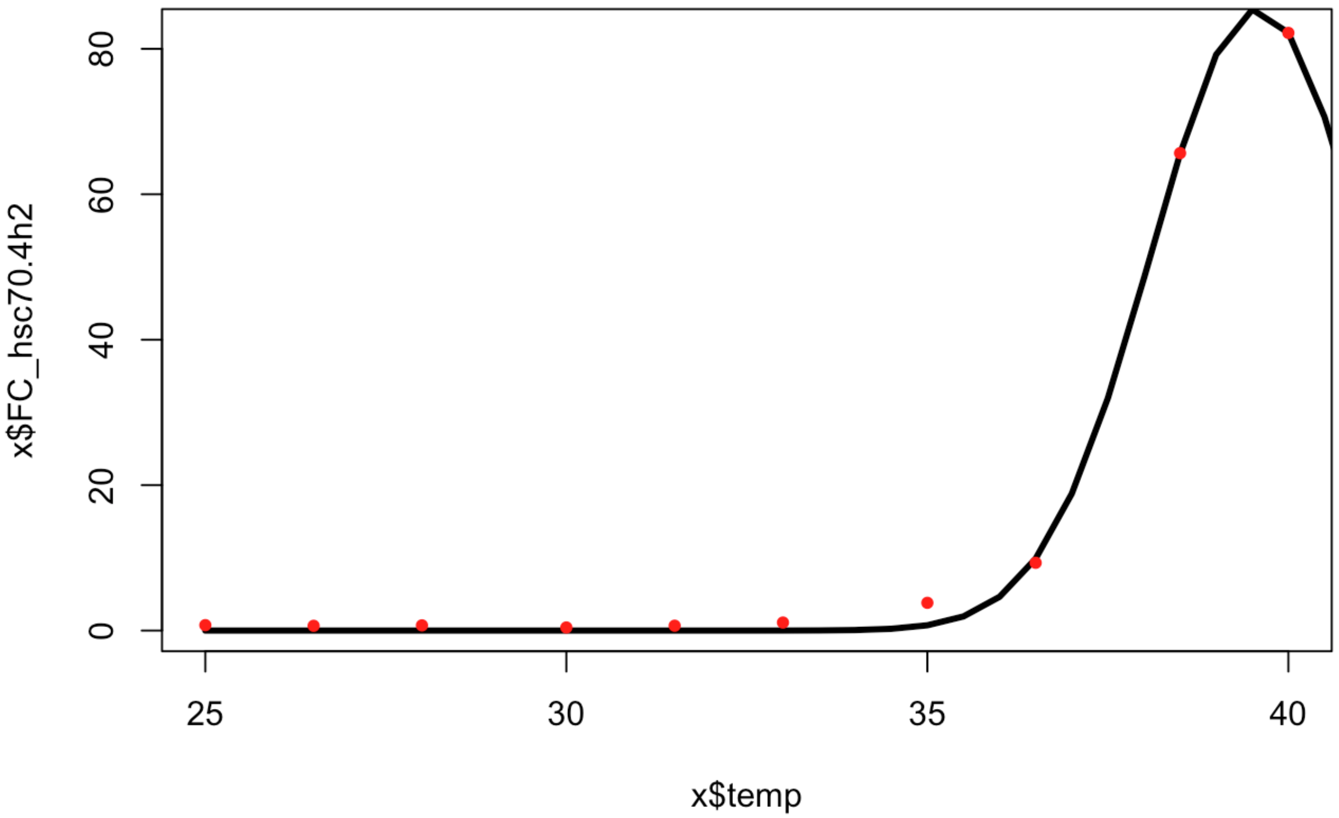
```
#sarah diamond's sample function
#Gaussian curve
G<-nls(FC_hsc70.4h2 ~ a*exp(-0.5*((temp-b)/c)^2),data=x, start=list(a=25, b=40,
c=1.5), trace=TRUE,control=nls.control(warnOnly = TRUE, tol = 1e-05, maxiter=100
0))
```

```
## 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
```

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


x$FC_hsc70.4h2
0
20
40
60
80
25
30
35
40
x$temp

```
#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.control(warnOnly = TRUE, tol = 1e-05, maxiter=1000))

#nls(hsc4.2 ~ c*((Temp/a)^(b-1))*exp(-(Temp/a)^b), start = list(a = 30, b = 30,c = 1), data = x,trace=TRUE,control=nls.control(warnOnly = TRUE, tol = 1e-05, maxiter=1000))

#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+(c-1)/(1+exp((b-temp)/a))),data=x, start=list(c=80,b=35,a=1.05), trace=TRUE,control=nls.control(warnOnly = TRUE, tol = 1e-05, maxiter=1000))
```

```
## 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 + (c - 1)/(1 + exp((b - temp)/a)))
##
## Parameters:
##   Estimate Std. Error t value Pr(>|t|)
## c 84.06200    1.06880   78.65 1.41e-11 ***
## b 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
```

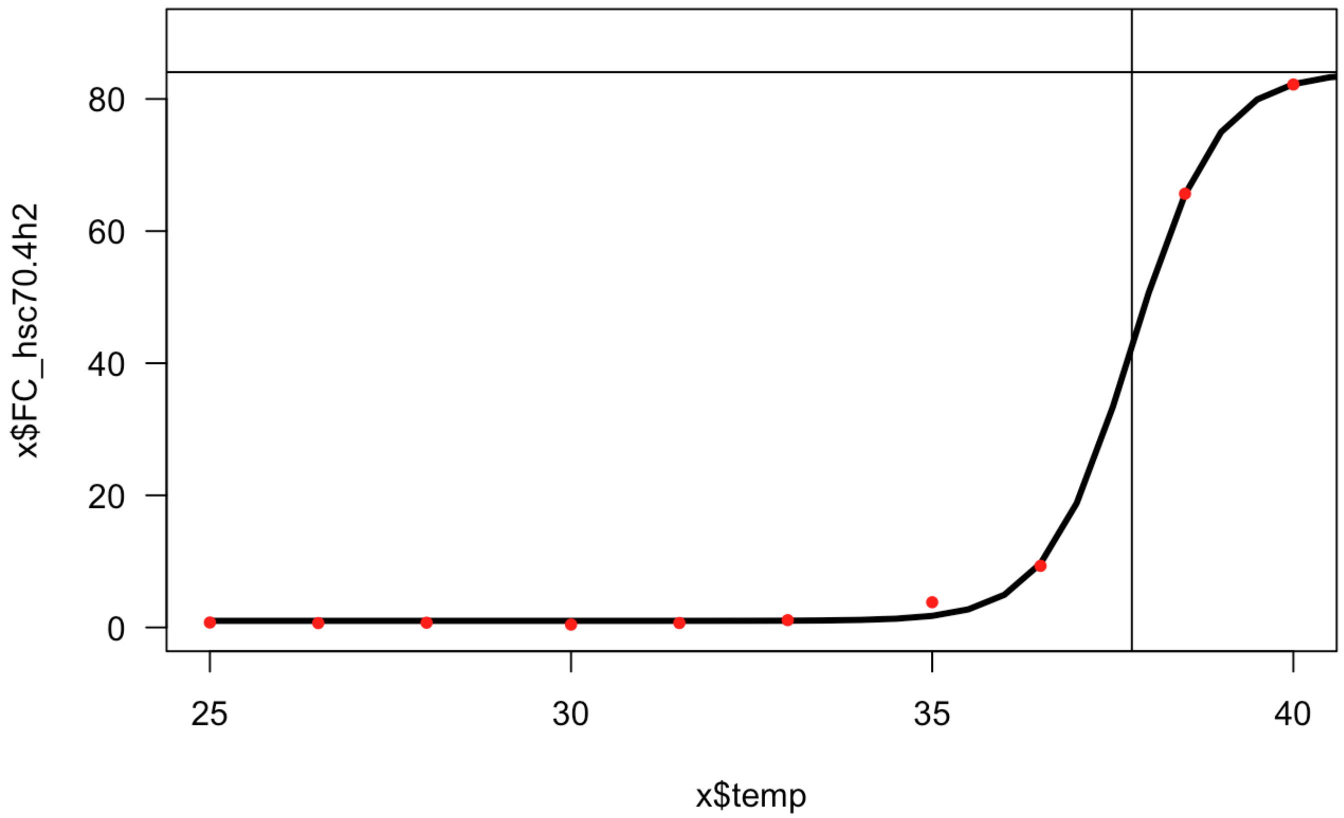
```
summary(B)$parameters
```

```
##     Estimate Std. Error   t value     Pr(>|t|)
## c 84.0619983 1.06879870  78.65092 1.413581e-11
## b 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])
```



```
#deriv(B)
#deriv(x$hsc4.2 ~ 1+(c-1)/(1+exp(b-Temp/a)), c("c","b","a"), func = TRUE)
#B2<-nls(hsc4.2 ~ (d+(c-d)/(1+exp(b-Temp/a))),data=x, start=list(c=40,b=3
5,a=1.1,d=1.1), trace=TRUE,control=nls.control(warnOnly = TRUE, tol = 1e-05, maxit
er=1000))
#lines(new,predict(B2,list(Temp=new)),lwd=3)

AIC(G,B)
```

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

grabbing value at inflection point and max

```
summary(B)$parameters[1:2,1]
```

```
##        c        b 
## 84.06200 37.76448
```

```
summary(B)$parameters[3,1]
```

```
## [1] 0.5884874
```

```
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] 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
```