

Introduction to Time-Series Analysis with PI System and R

Prof. Brian D. Davison

Prof. Martin Takáč



Presentation Outline

- Preliminaries: Sensors, Data set, and Sampling
- Task #1: Prediction of ADF
- Task #2: Modeling Cooling
- Task #3: PCA for Visualization
- Lessons Learned

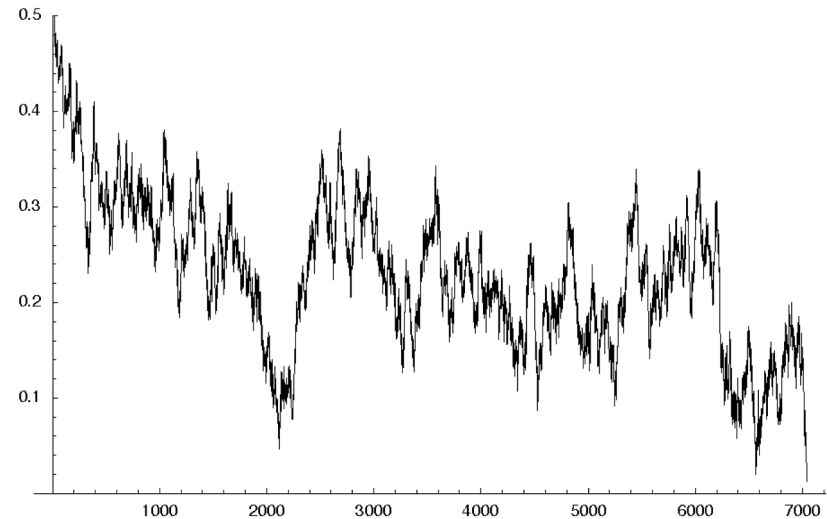
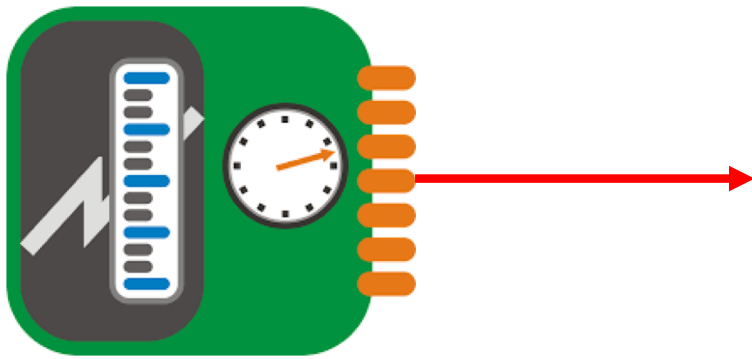


Preliminaries

Sensors, Data set, and Sampling

Sensor data

- frequent readings of sensor values (e.g., each second/minute)



- Common problems:
- missing values (wrong reading from sensor/communication issue)
 - a sensor fails and gives wrong readings

Brewery Dataset

Temperature sensors

top

middle

bottom

Pressure sensor

.... and many more sensors...
and even more fermenters...

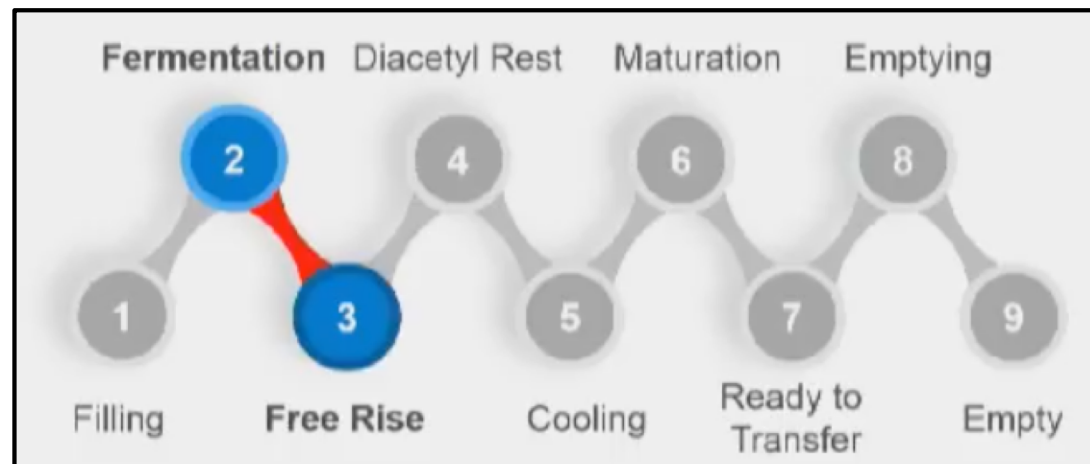


Fermenter

source: wikipedia

Brewery Dataset

- From a company that produces more than two dozen craft beers
- Different beers can use different yeasts during fermentation and different fermenting temperatures



From OSIsoft Users Conference 2017

OSIsoft: Introduction to Process Optimization and Big-Data Analytics with the PI System
<https://www.youtube.com/watch?v=9AdvKkyTeBE>

Stages of beer fermentation

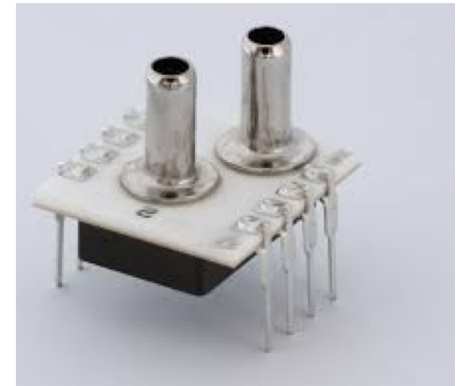
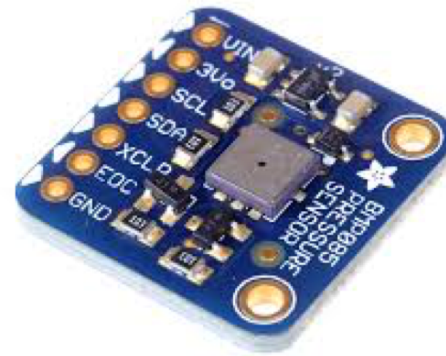
Formally, a time series

- is an **ordered sequence** of values of a variable at **equally spaced time intervals**
- can be built on top of the data obtained from sensors
 - choose the size of time interval (e.g., 1 hour)
 - represent all measurements in each hour by one value (e.g., average value / first value / last value ...)
 - we obtain a collection of values (e.g., one number per hour)
 - the PI system can extract such values from collected data
- has many applications in forecasting, monitoring or even feedback and feed-forward control

Multiple sources and “sampling” frequencies

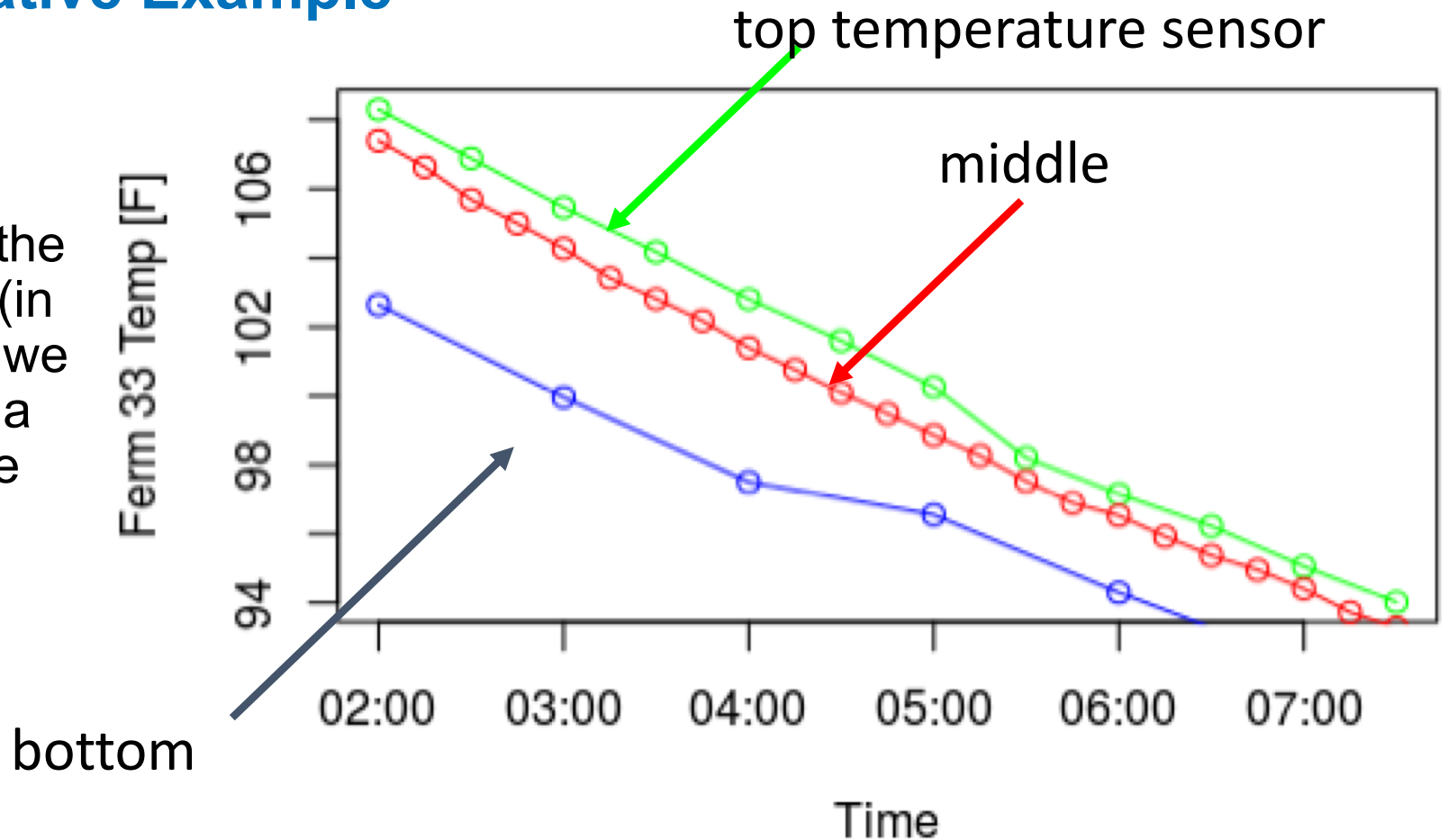
Real life problem = multiple sensors

Manual = infrequent



An illustrative Example

Before doing analytics, we need to align the data streams (in this example, we could choose a multiple of one hour)!





Task #1: Prediction of ADF

The Problem Setting

- Automated measuring of gravity of wort is expensive
- The brewery is hence measuring it manually (and infrequently)
- It is used to compute **Apparent Degree of Fermentation (ADF)** which is an important indicator of the status of fermentation
(Note: ADF indicates how much sugar has been converted to alcohol)

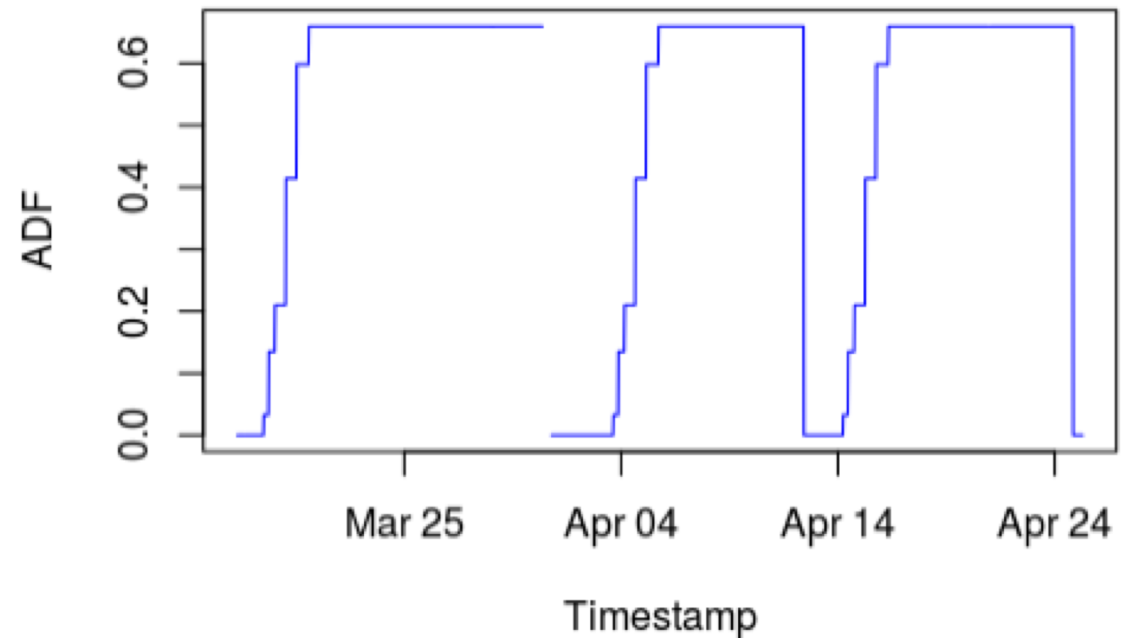
Our task: Predict ADF as well as possible (and eliminate a lot of manual work and improve quality of beer, decrease cost of fermentation)

Data and Challenges

- We can use the data from all sensors to predict ADF

BUT

- We have only a few ADF values per fermentation (and moreover it is **not computed in “regular” intervals**)



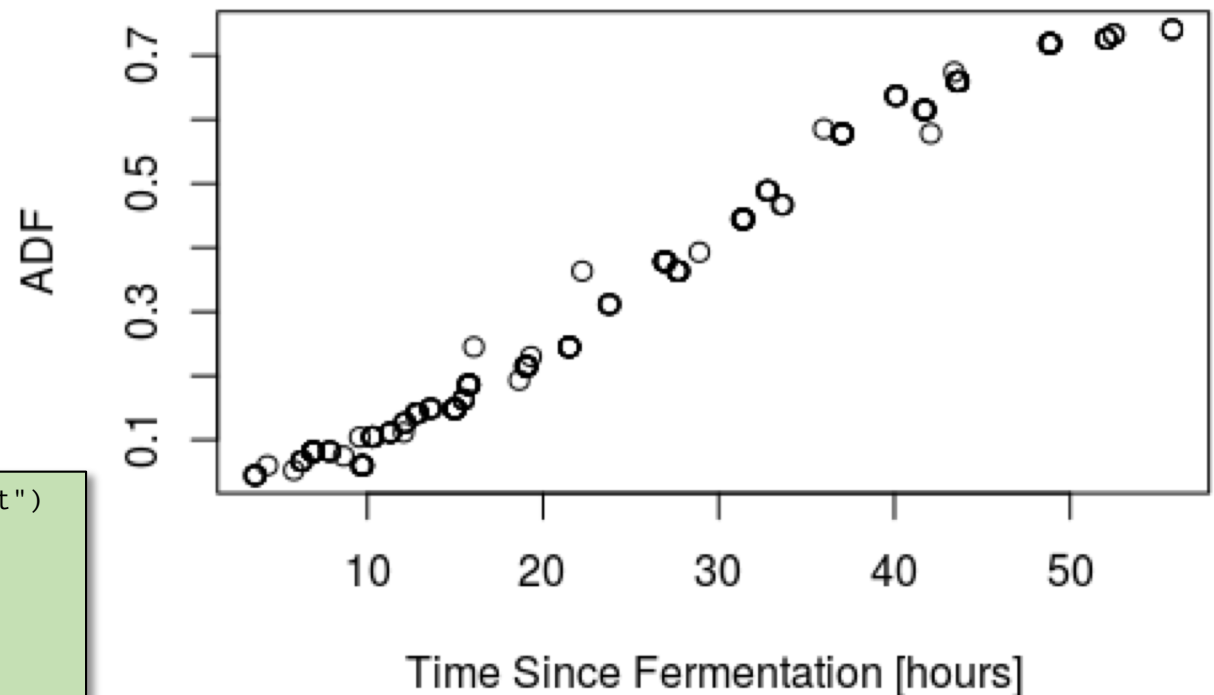
The First Approach

- Let's fix one beer brand, and plot our ADF points:

x-axis - Time
from start of
fermentation
y-axis - ADF

```
MyData <- read.csv(file="regression.txt")
MyData$time <- MyData$time/60/60
plot(MyData$time, MyData$adf, xlab =
"Time Since Fermentation [hours]",
ylab = "ADF", main = "Data without
outliers")
```

Data without outliers



Simple Linear Regression Model

$$Y = \alpha + \beta X + \epsilon$$

intercept

slope

errors of the response variables should be uncorrelated with each other

response (dependent) variable

explanatory (independent) variable

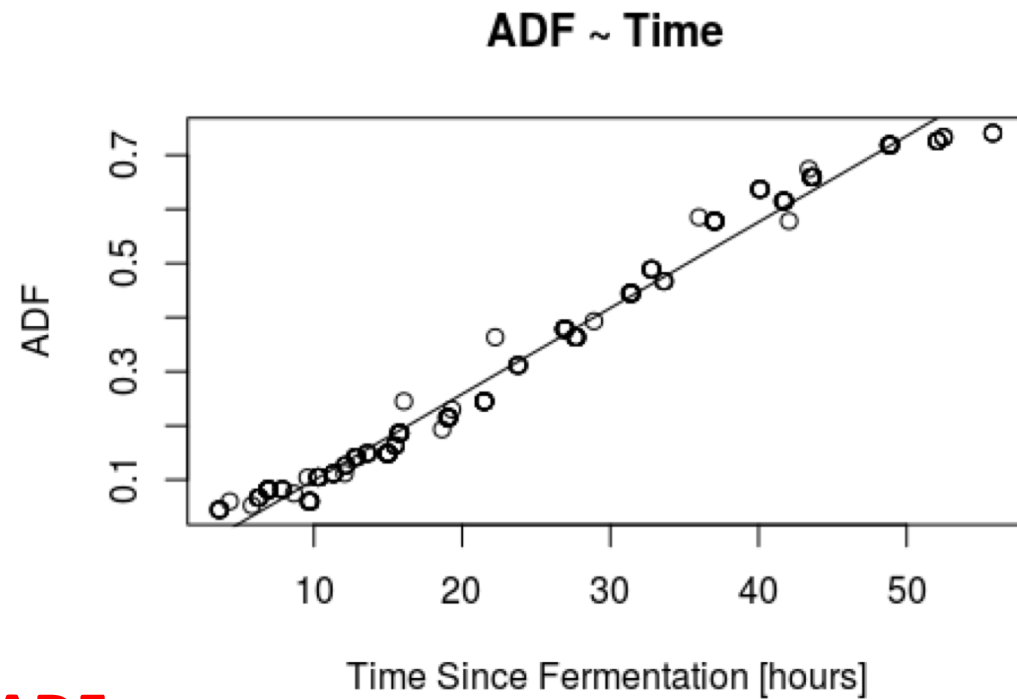
Linear Fit

Estimate

(Intercept) -0.0598497

Time 0.0159091

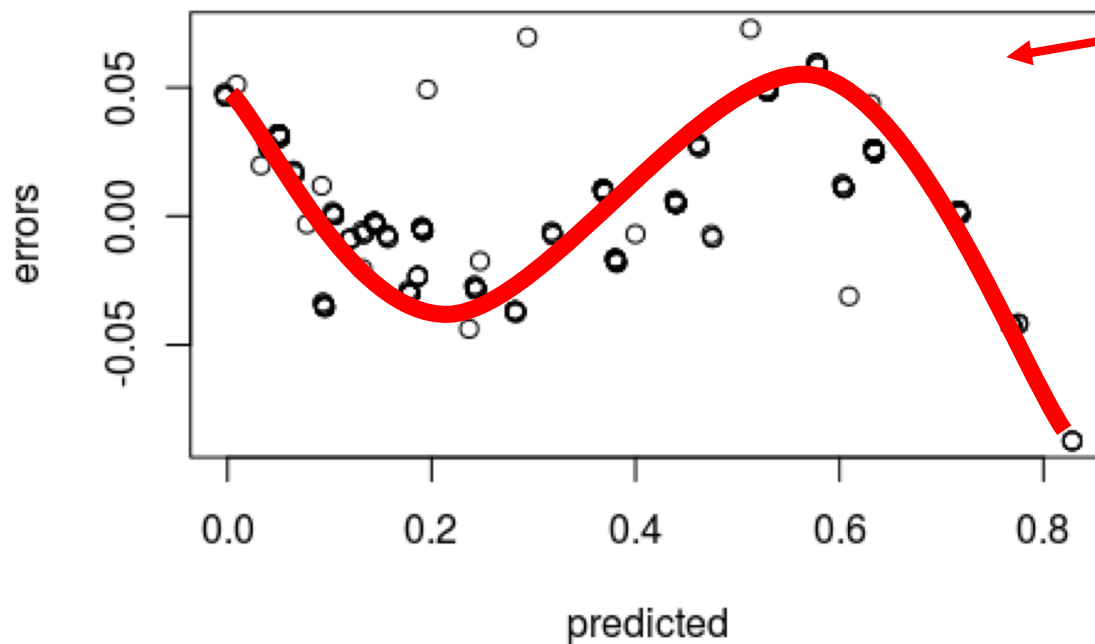
**each hour, ADF
increases by ~0.016**



```
plot(FilteredTimes, FilteredADF, xlab =  
  "Time Since Fermentation [hours]",  
  ylab='ADF', main="ADF ~ Time")  
linMod <- lm(FilteredADF ~ FilteredTimes)  
abline(linMod$coefficients)  
  
summary(linMod)
```

Checking Assumptions with a Residual Plot

Residuals vs. predicted ADF



errors should be independent of predicted values

Our model is **NOT** ok!

```
predicted <- fitted(linMod)
errors <- resid(linMod)
plot(predicted, errors, main = "Residuals
vs. predicted ADF")
```


What was wrong? How to improve it?

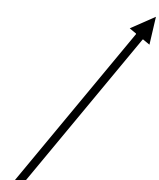
- We tried our luck with a simple linear model
- The model works OK for the data in the middle of our range, but is failing for small and large values of ADF

- **As data scientists, we need to understand our problem domain.**
- We need to learn about beer fermentation!

Understanding the Ferme

- What influences the speed of fermentation?
 - temperature (*we are cool*)
 - amount of yeast
 - amount of sugar and ethanol

Gilliland, R.B., 1962. Yeast reproduction during fermentation. *Journal of the Institute of Brewing*, 68(3), pp.271-275.



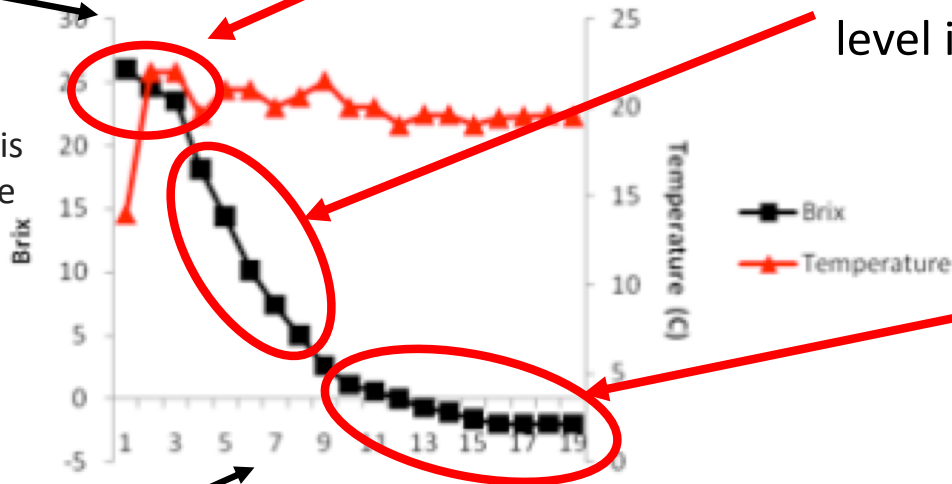
INTRODUCTION

IN normal brewery fermentations, the yeast first undergoes a lag phase during which little growth takes place; then there is a growth phase, during which the yeast reproduces fairly rapidly; and finally comes the fermentation phase, during which yeast growth gradually slows down and the wort is fermented. A vigorous growth phase is essential in order to obtain an adequate speed of fermentation and a low final gravity; but brewers do not want more yeast growth than is necessary to produce these results, as excess yeast cannot be sold at an economic price when it has been grown with duty-paid worts as a source of carbohydrate. In addition, increased yeast production involves extra cost in separation of the yeast from its associated barm beer and extra wastage of beer. The effect of different factors on the

Speed of Fermentation

sugar content

One degree Brix is 1 gram of sucrose in 100 grams of solution



cells are adapting to the wort conditions and engaging in cell division

fastest rate of fermentation, ethanol level is relatively low

The rate slows because ethanol in the medium forces an adaptation of the plasma membrane. The yeast can easily form a membrane that depends upon ethanol replacing water for its structure and functionality...

time

http://wineserver.ucdavis.edu/industry/enology/fermentation_management/wine/problem_fermentations.html

New (improved) model

- We now understand that there are different rates of fermentation.
- How about fitting 3 linear models? We would also need to figure out the break-points.

$$Y = \begin{cases} \alpha_1 + \beta_1 X + \epsilon, & X \leq \gamma_1 \\ \alpha_2 + \beta_2 X + \epsilon, & \gamma_1 \leq X \leq \gamma_2 \\ \alpha_3 + \beta_3 X + \epsilon, & \gamma_2 \leq X \end{cases}$$

parameters to estimate

Fitted Model

intercept above zero

decrease of slope. New slope is **0.003764**

Meaningful coefficients of the linear terms:

(Intercept)	x	U1.x	U2.x
0.005924	0.009560	0.008451	-0.014247

Estimated Break-Point(s):

psi1.x	psi2.x
14.57	45.61

increase of slope. New slope is **0.018011**

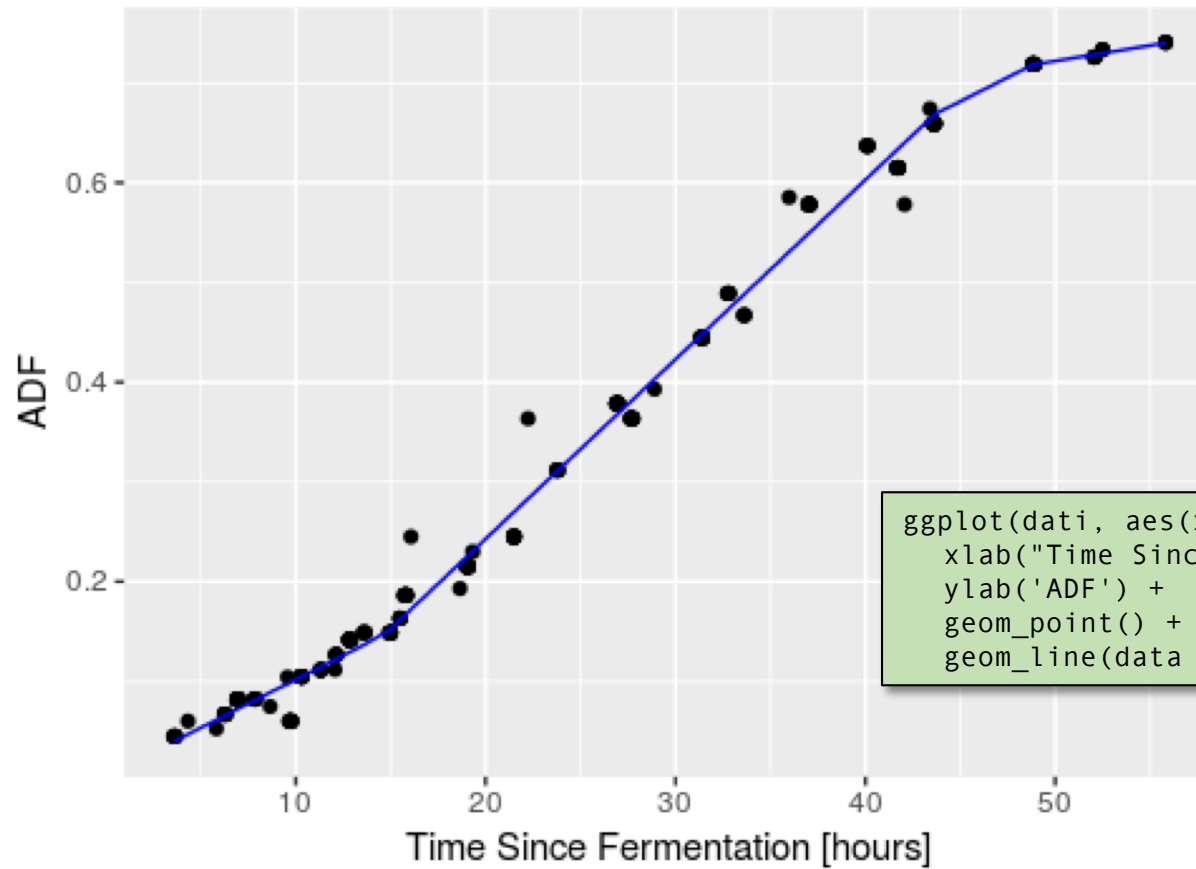
initial slope

break-points

```
xx <- FilteredTimes
yy <- FilteredADF

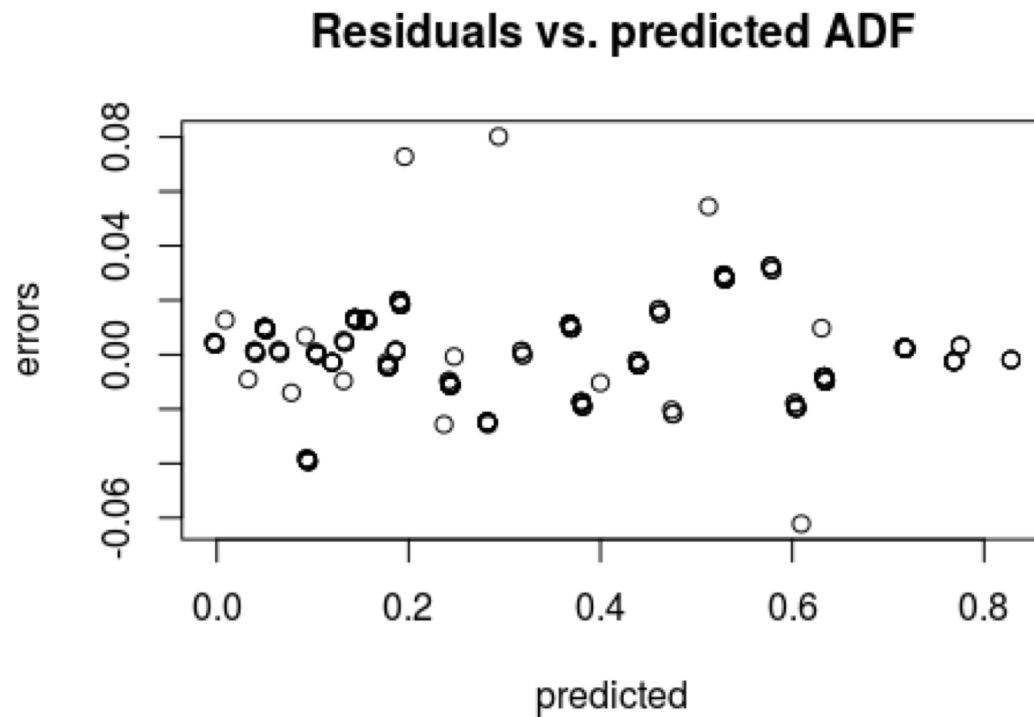
dati <- data.frame(x = xx, y = yy )
out.lm <- lm(y ~ x, data = dati)
o <- segmented(out.lm, seg.Z = ~x,
  psi = list(x = c(10,40)),
  control = seg.control(display = FALSE))
dat2 = data.frame(x = xx, y =
  broken.line(o)$fit)
o
```

Fitted Model



```
ggplot(dati, aes(x = x, y = y)) +  
  xlab("Time Since Fermentation [hours]") +  
  ylab('ADF') +  
  geom_point() +  
  geom_line(data = dat2, color = 'blue')
```

Checking Assumptions with a Residual Plot



```
errors <- resid(o)  
plot( predicted, errors, main =  
      "Residuals vs. predicted ADF" )
```

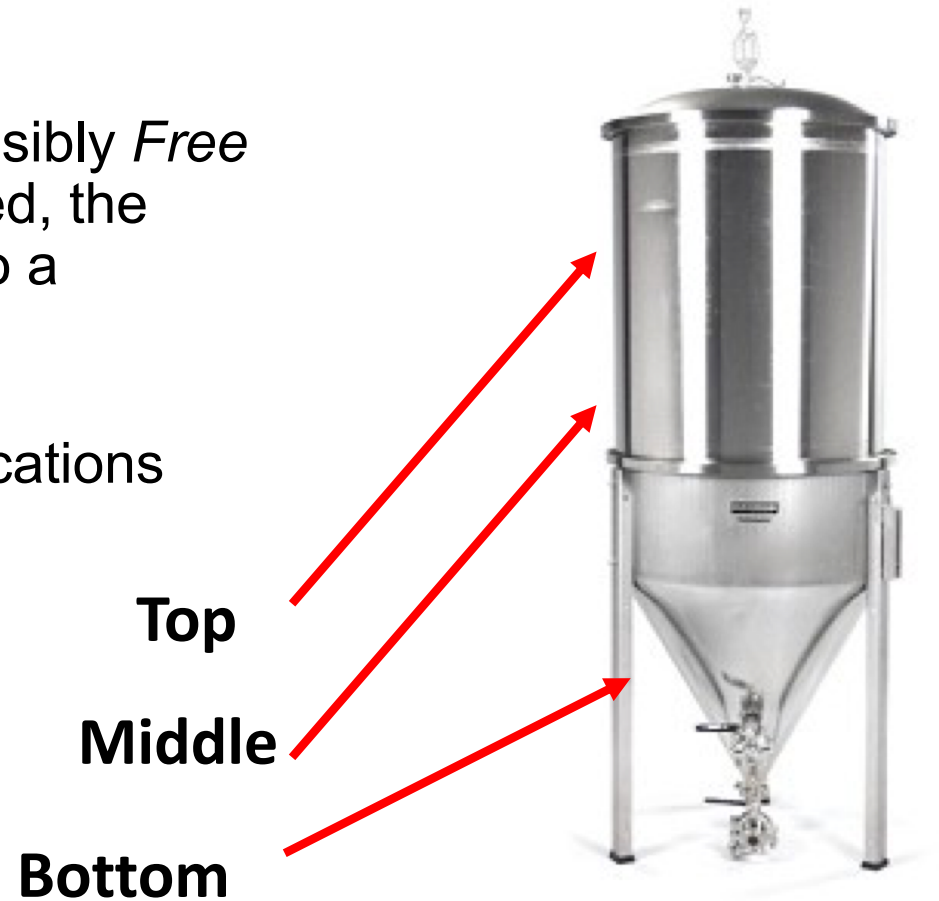


Task #2

Cooling prediction

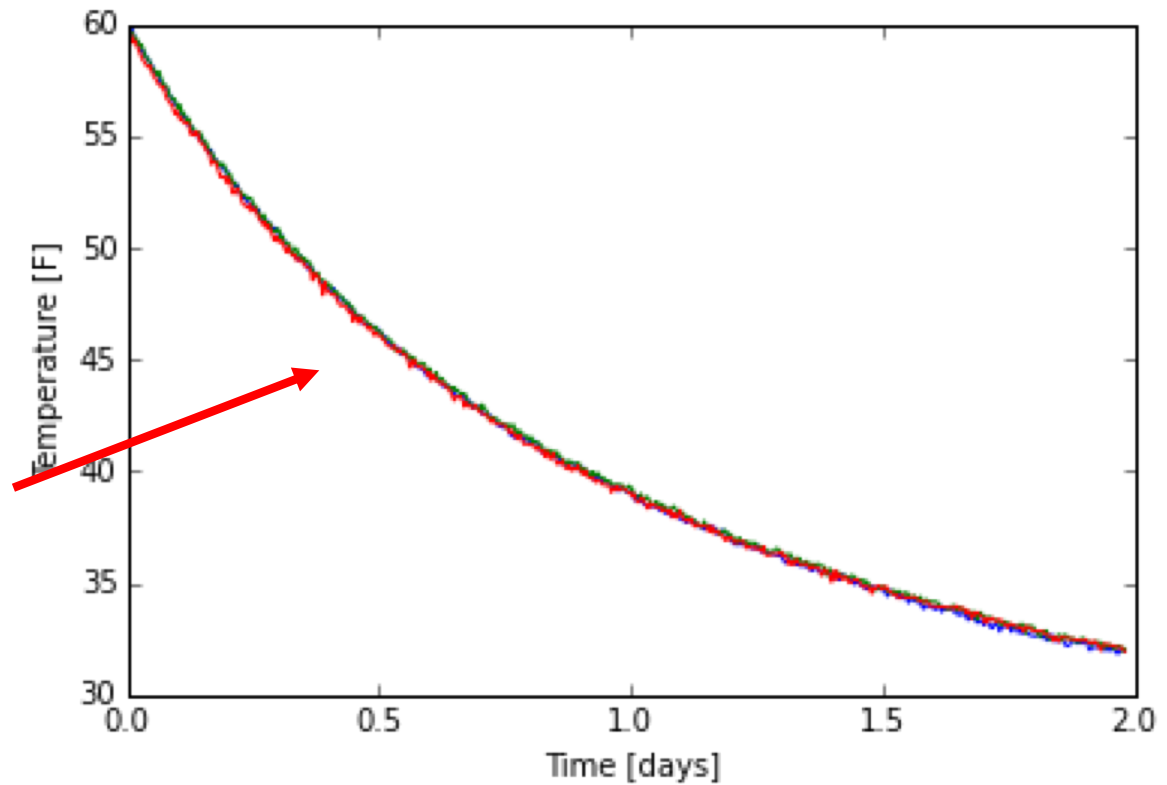
The Problem Setting

- After the fermentation (and possibly *Free Rise* or *Diacetyl Rest*) is finished, the beer has to be cooled almost to a freezing point
- The data contains temperature measurements at 3 different locations

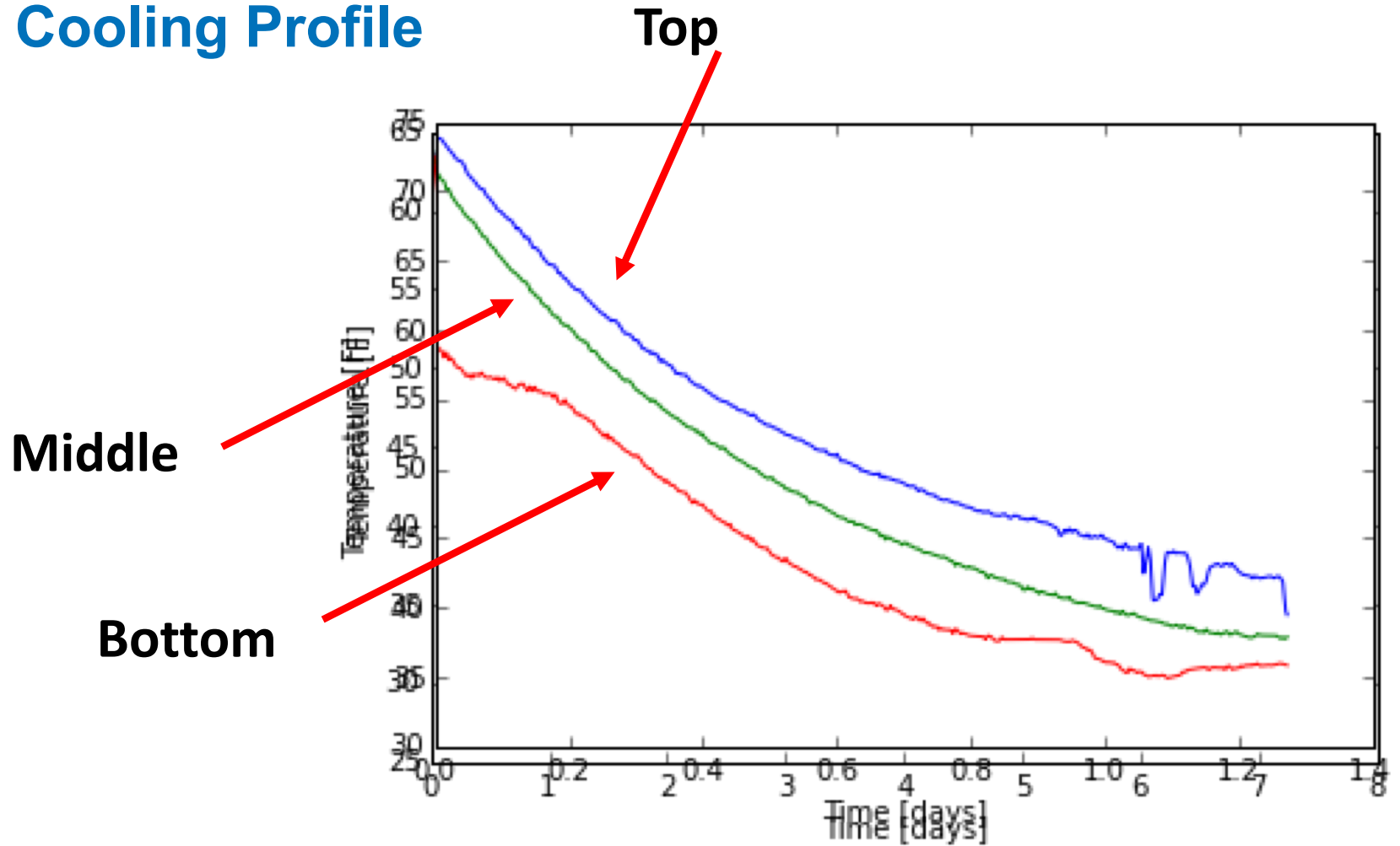


Typical Cooling Profile

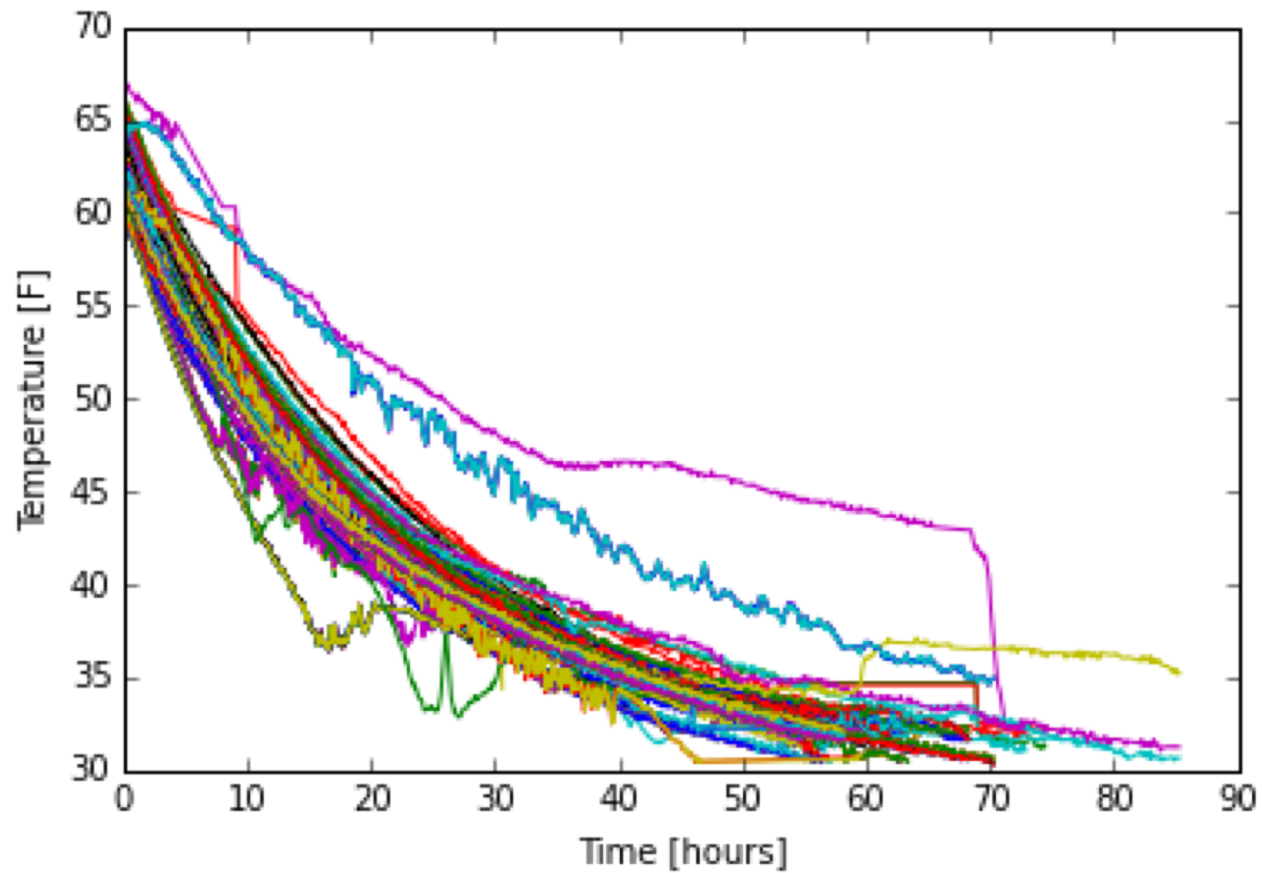
In the ideal case, the beer is cooling nicely and all sensors read similar values



Atypical Cooling Profile

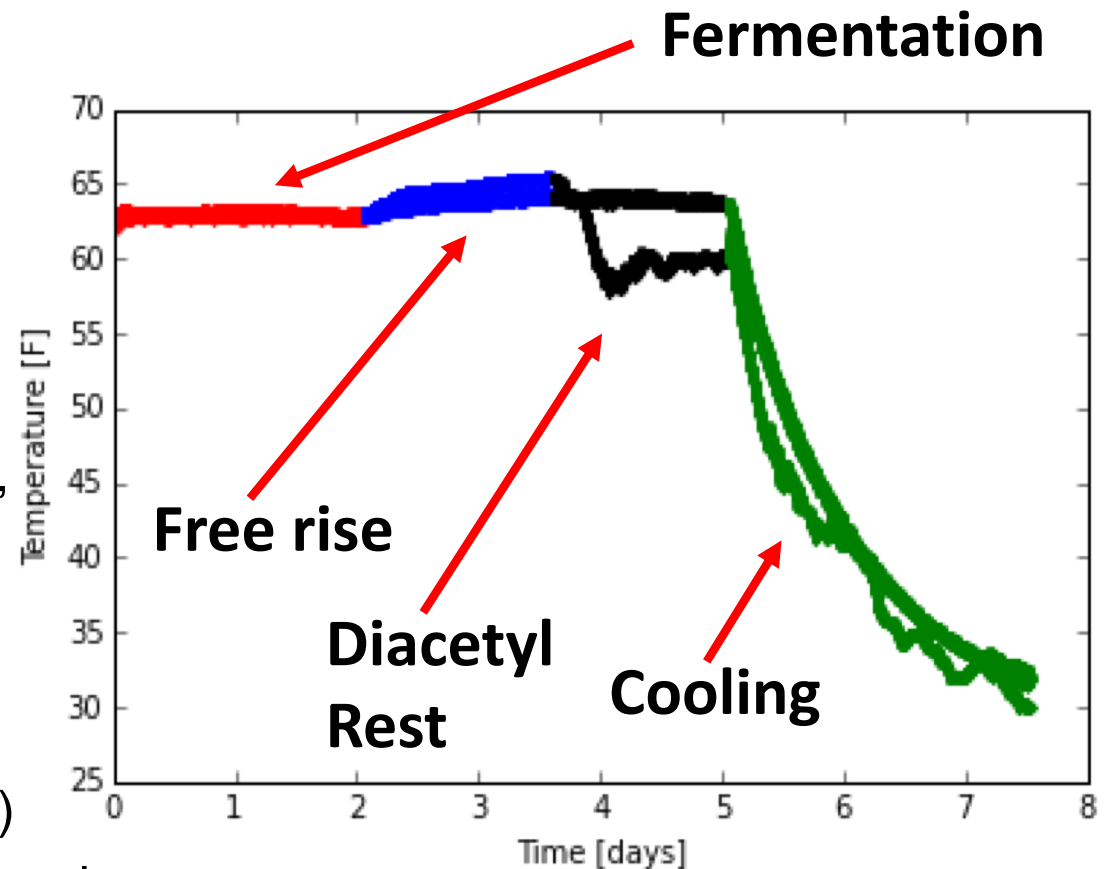


All together

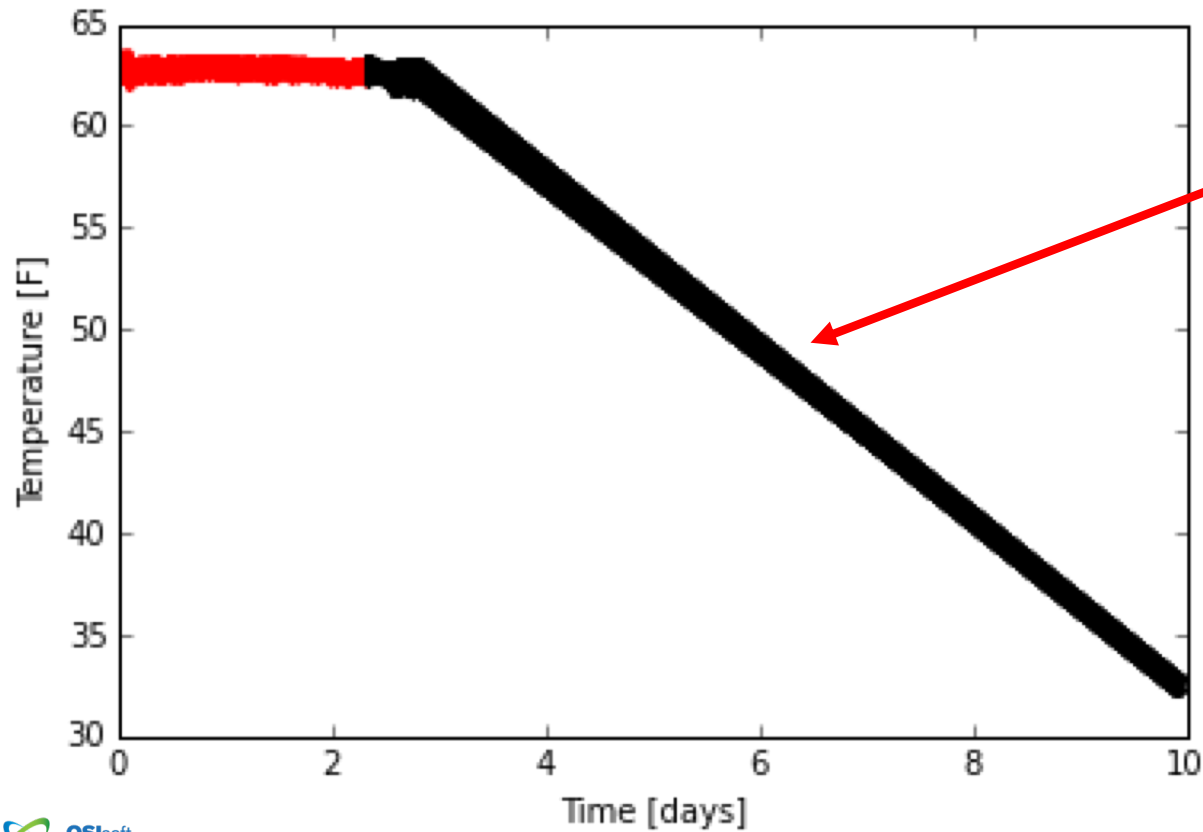


Issues with data

- For each time point, we know in which state of fermentation the beer is (colors on graph)
- We also know about other sensors, such as the cooling valves
- During fermentation, we try to keep temperature constant
- The cooling is achieved by three cooling valves (usually open 100%)
- However, there are many inconsistencies

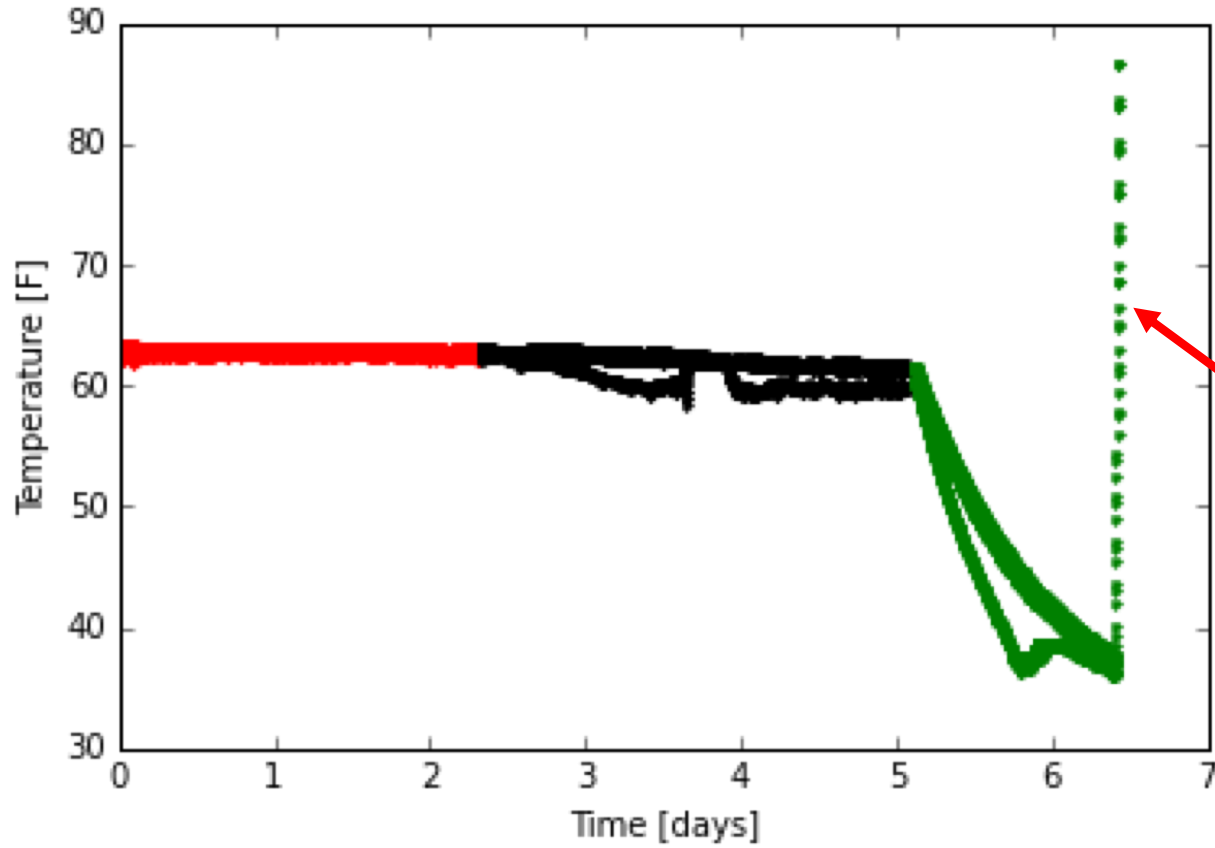


Issues with data

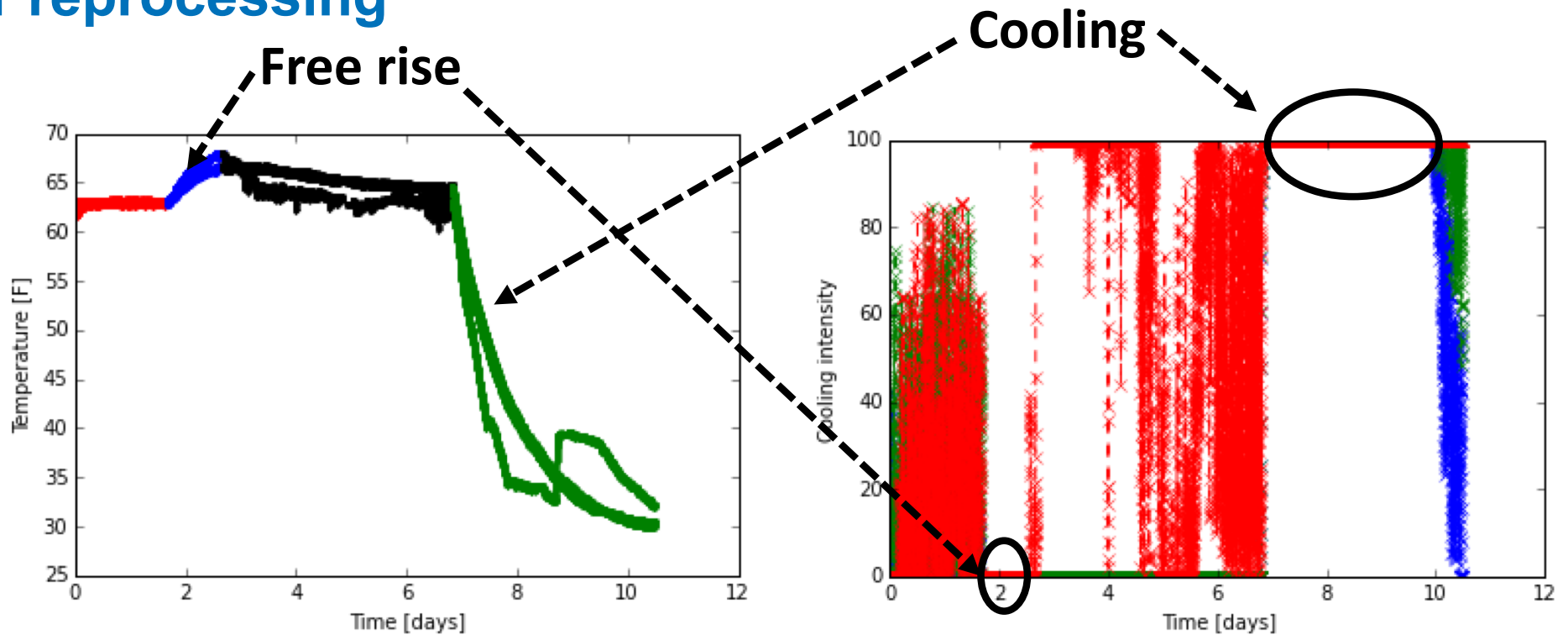


**Cooling
labeled as Diacetyl
Rest**

Issues with data



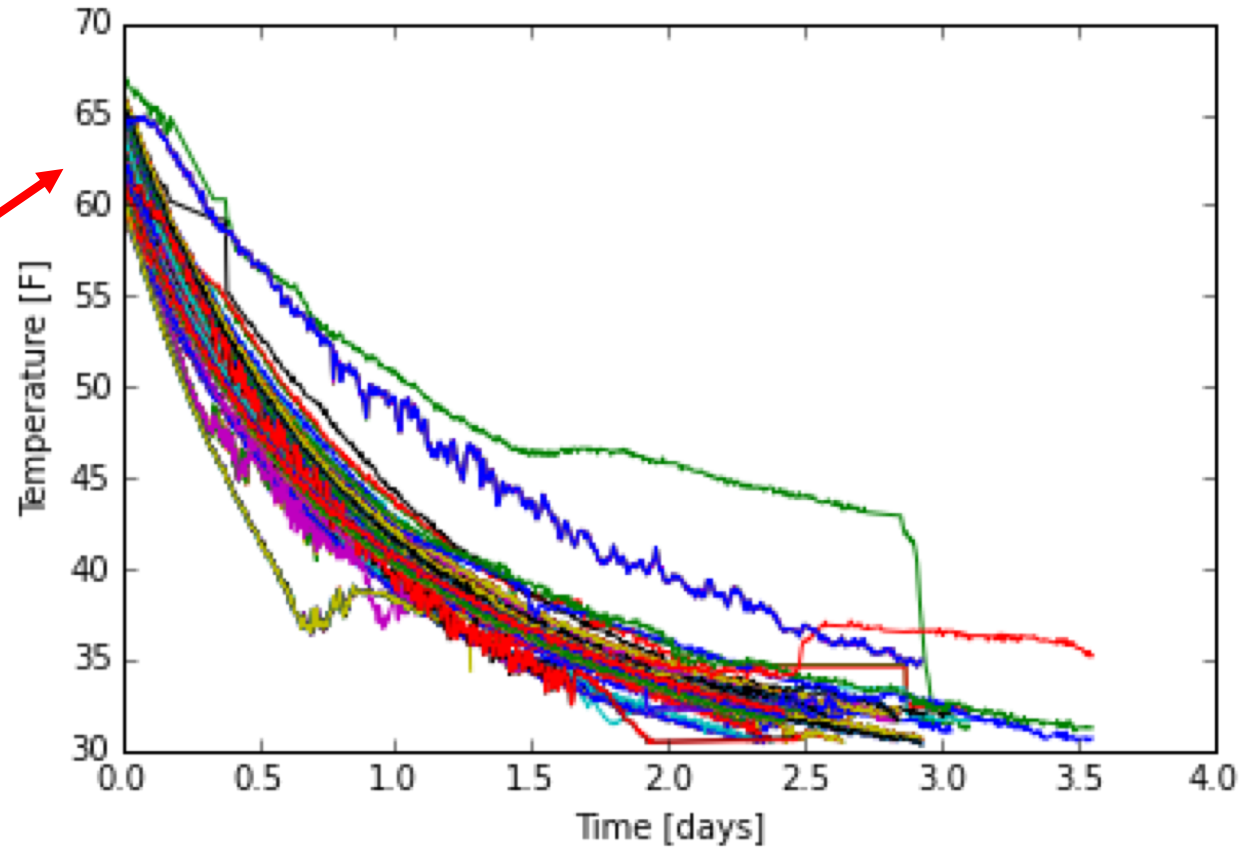
Preprocessing



- We will ignore the “label” of the fermentation “stage”
- *We define a cooling period simply as a uninterrupted interval with 100% cooling capacity*
- Moreover, we will be interested in the data until the temperature reaches ~32F

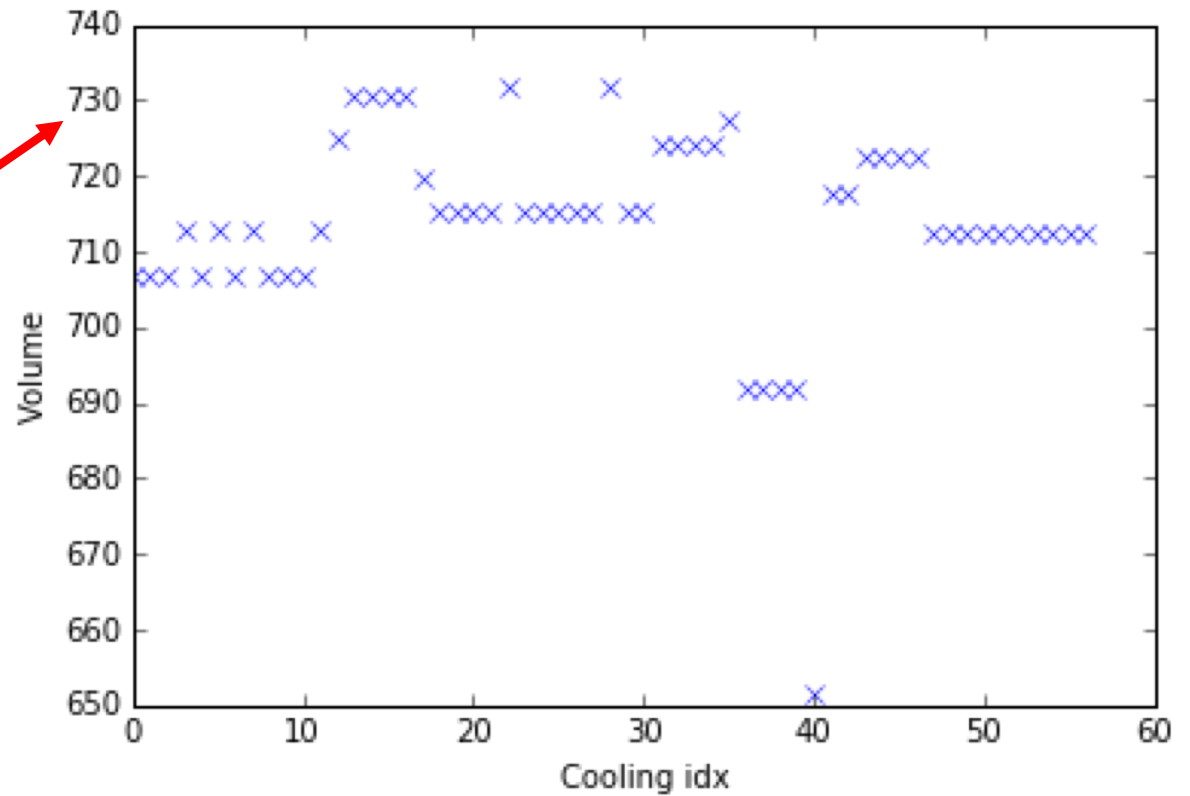
Filtered cooling profiles

- different initial temperature for various coolings



Volumes

- various volumes of beer in fermenter



Thermodynamic heat transfer

- how the temperature T_t will change in a small time interval Δt ?
- real answer is hard and involves solving PDEs,
- we can try to approximate it

$$T_{t+1} - T_t \approx \frac{\alpha(T_t - \beta)}{Vol}$$

Annotations for the equation:

- T_{t+1} : New temperature
- T_t : Old temperature
- $\alpha(T_t - \beta)$: Cooling Rate (<0)
- β : Temperature of "cooling" substance
- Vol : Volume

Some simplifications

$$T_{t+1} - T_t \approx \frac{\alpha(T_t - \beta)}{V_{ol}}$$



$$T_{t+1} \approx \left(1 + \frac{\alpha}{V_{ol}}\right)T_t - \frac{\alpha\beta}{V_{ol}}$$

Time difference $\Delta t = 30$ min

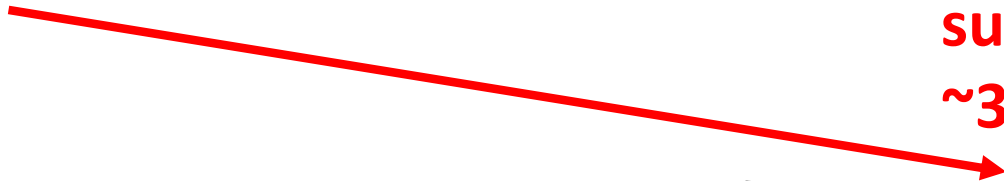
Parameters:

	Estimate	Std. Error	t value	Pr(> t)
alpha	-17.1918	0.3134	-54.86	<2e-16 ***
beta	30.4543	0.2427	125.49	<2e-16 ***

```
d <- read.csv(file="UC2_TS_30.dat")
yOld <- d$o1
y <- d$n1
vol <- d$v

m <- nls(y ~ (1+a/vol)*yOld - a*b/vol)
summary(m)
```

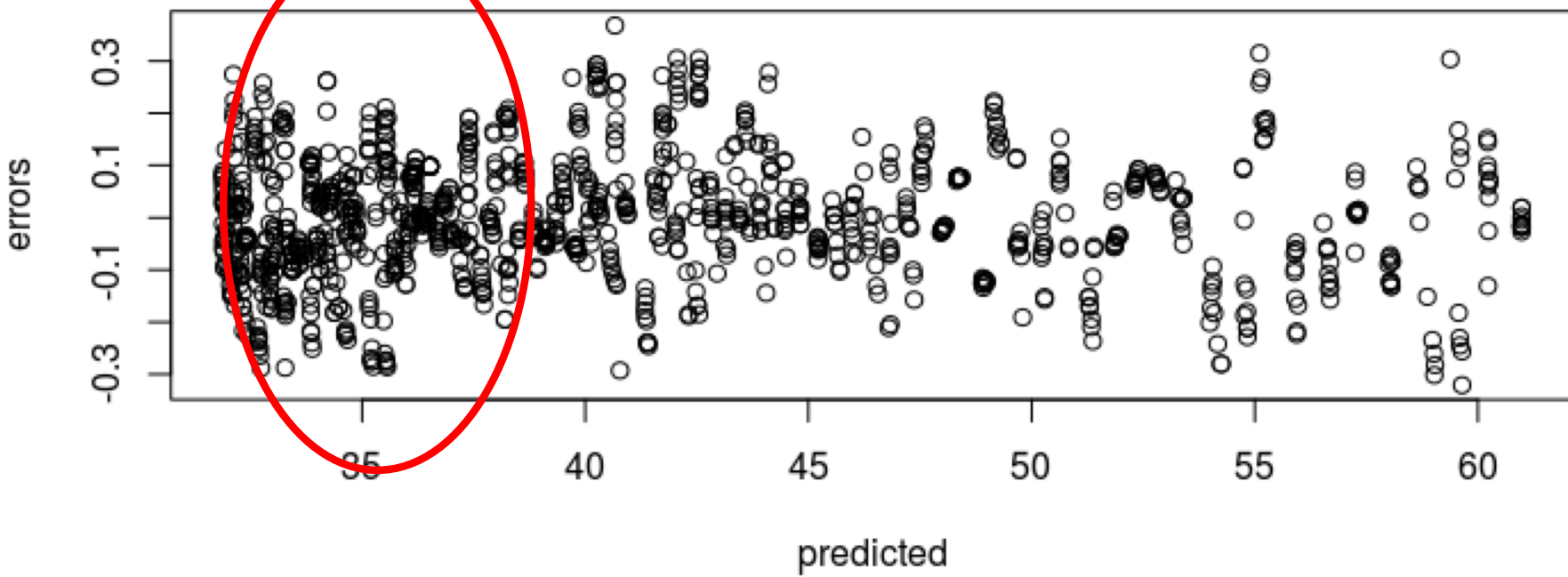
**The cooling
substance is
~30.4F**

$$T_{t+1} - T_t \approx \frac{\alpha(T_t - \beta)}{Vol}$$


Residuals

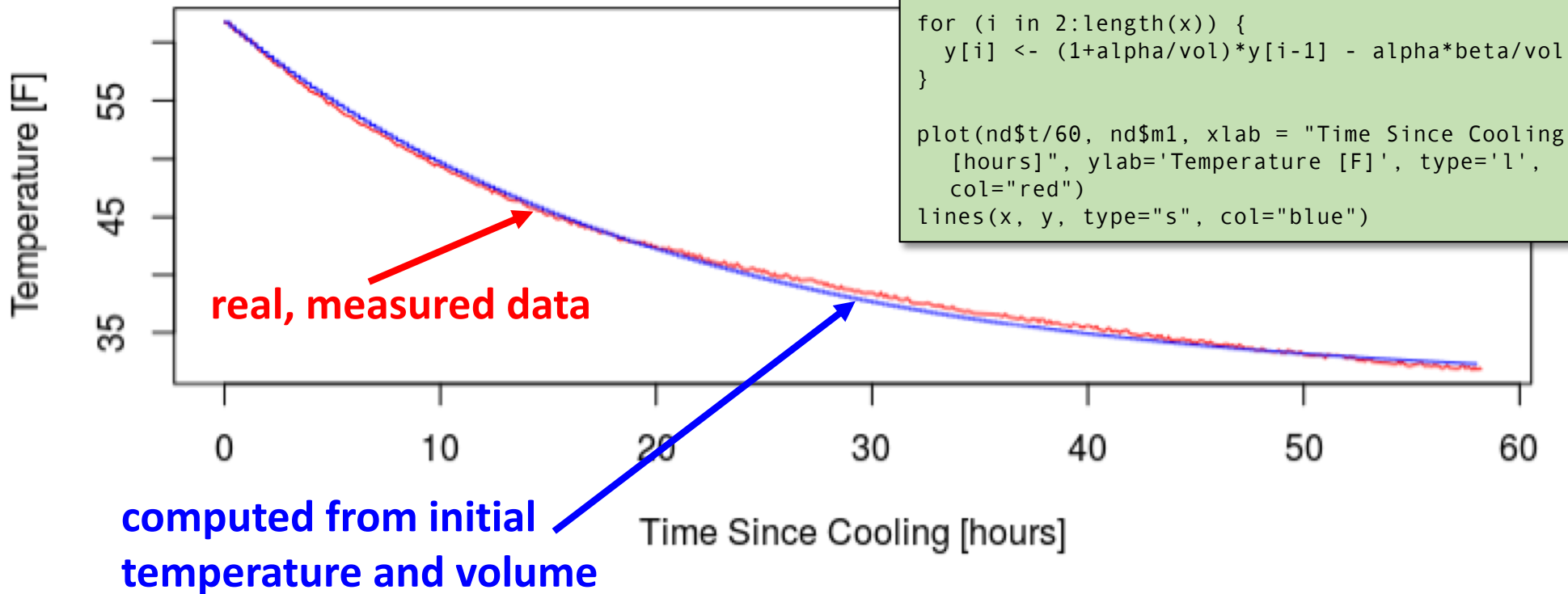
we have more data when it is cooler

Residuals vs. Predicted Temperature





```
predicted <- fitted(m)
errors <- resid(m)
plot(predicted, errors)
```

Real cooling profile vs. Predicted cooling profile



```
nd <- read.csv("UC2_Prediction.dat")  
  
x<-seq(0,58*4,1)*0.25; y<-x*0  
y[1] <- nd$m1[1]; vol <- nd$v[1]  
alpha <- summary(m)$coefficients[1]  
beta <- summary(m)$coefficients[2]  
  
for (i in 2:length(x)) {  
  y[i] <- (1+alpha/vol)*y[i-1] - alpha*beta/vol  
}  
  
plot(nd$t/60, nd$m1, xlab = "Time Since Cooling  
[hours]", ylab='Temperature [F]', type='l',  
col="red")  
lines(x, y, type="s", col="blue")
```

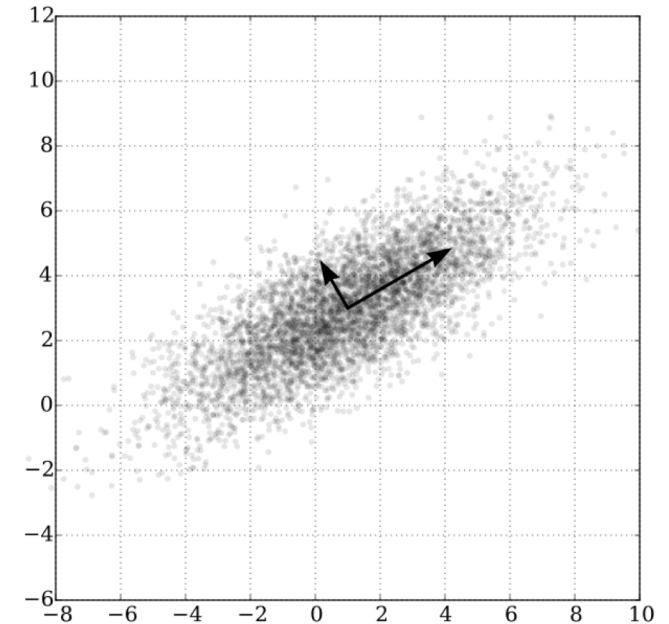


Task #3

Principal component analysis for visualization

Principal component analysis (PCA)

- PCA performs **dimensionality reduction** and de-correlation of dimensions
 - The “real” data can lie within a smaller dimensional subspace
- PCA is useful for **visualization** and pre-processing
 - Detect patterns and outliers
 - Produce fewer and uncorrelated dimensions
 - Good for data with relatively few labels



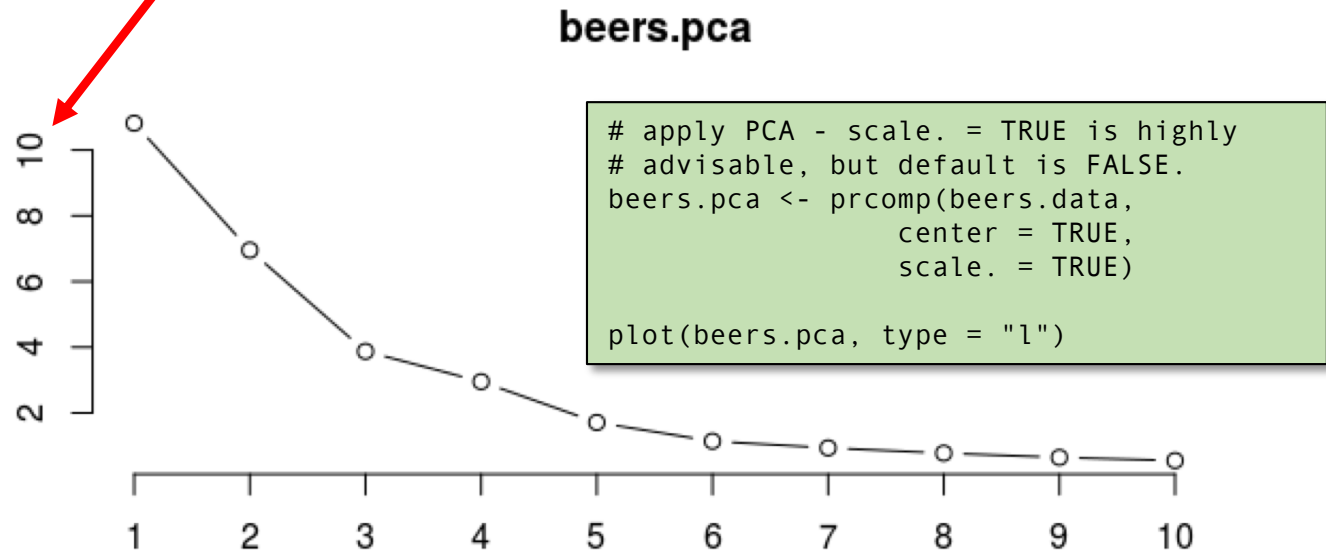
Data preprocessing

- We have a lot of data from sensors; it is a bit tricky to use PCA directly (too many points, missing data)
- More importantly, we want to see when **cycles** might be same or differ
- We represented each fermentation cycle by 31 features, like
 - highest ADF
 - mean/min/max temperature during fermentation
 - volume
 - mean/min/max temperature during cooling
 - duration of fermentation
 - duration of cooling
 -

Visualization in 3D

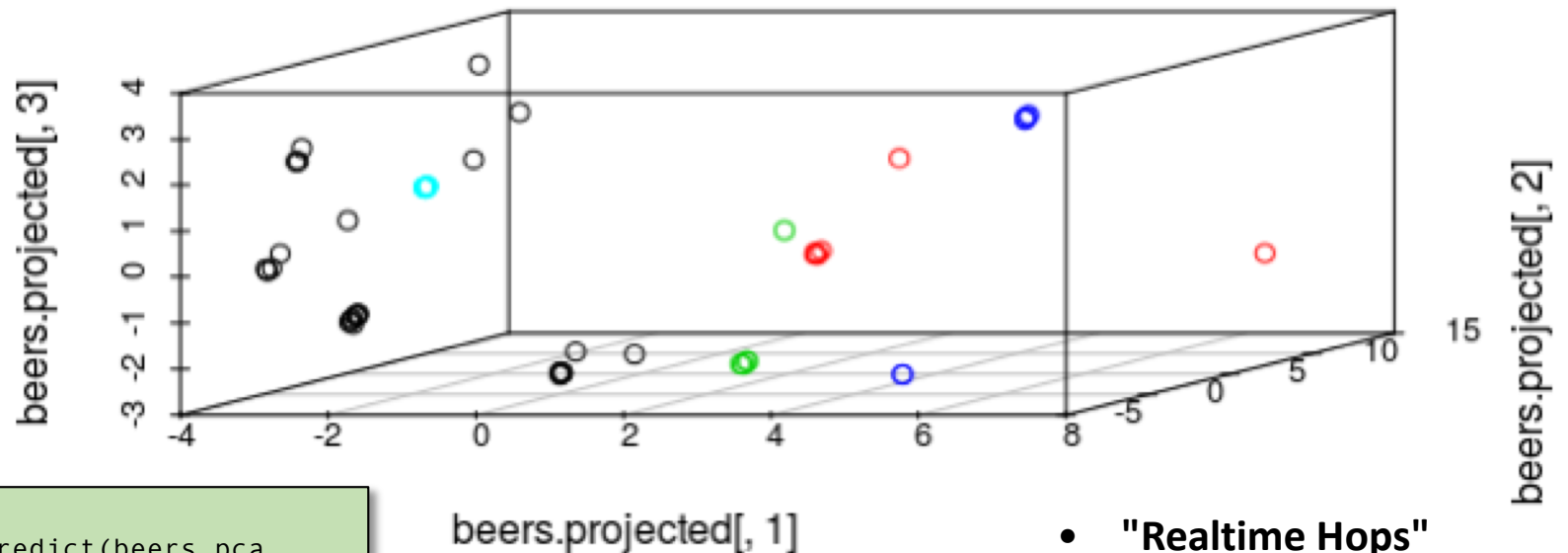
- We scale the data and compute the principal components
- We project the data onto two or three principal components that represent the most variance and plot them

how much variance is captured by the i^{th} principal component



Visualization in 3D

PCA

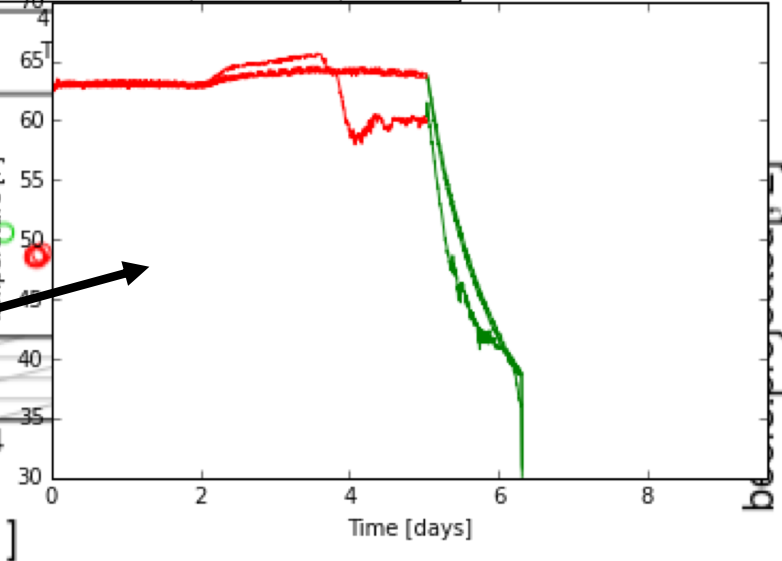
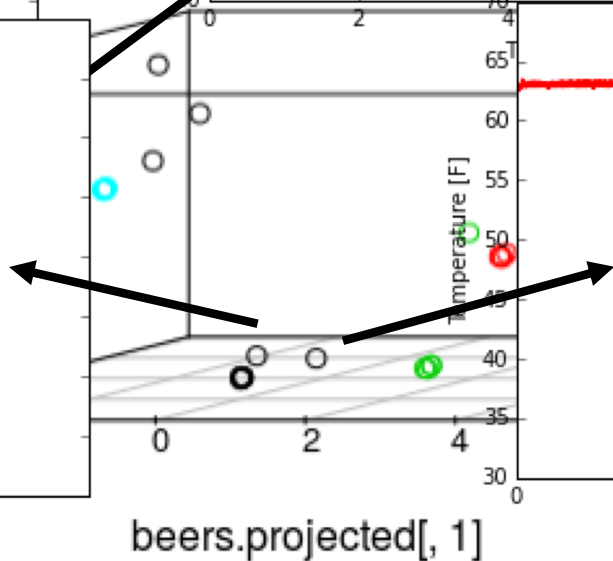
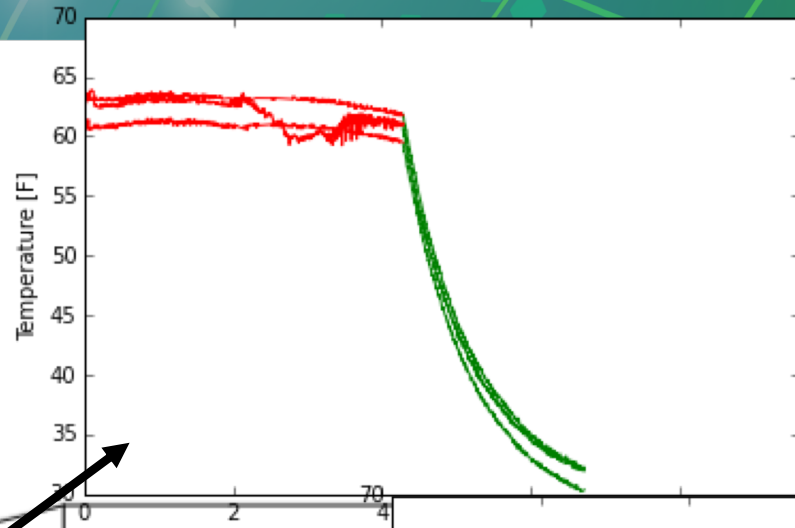
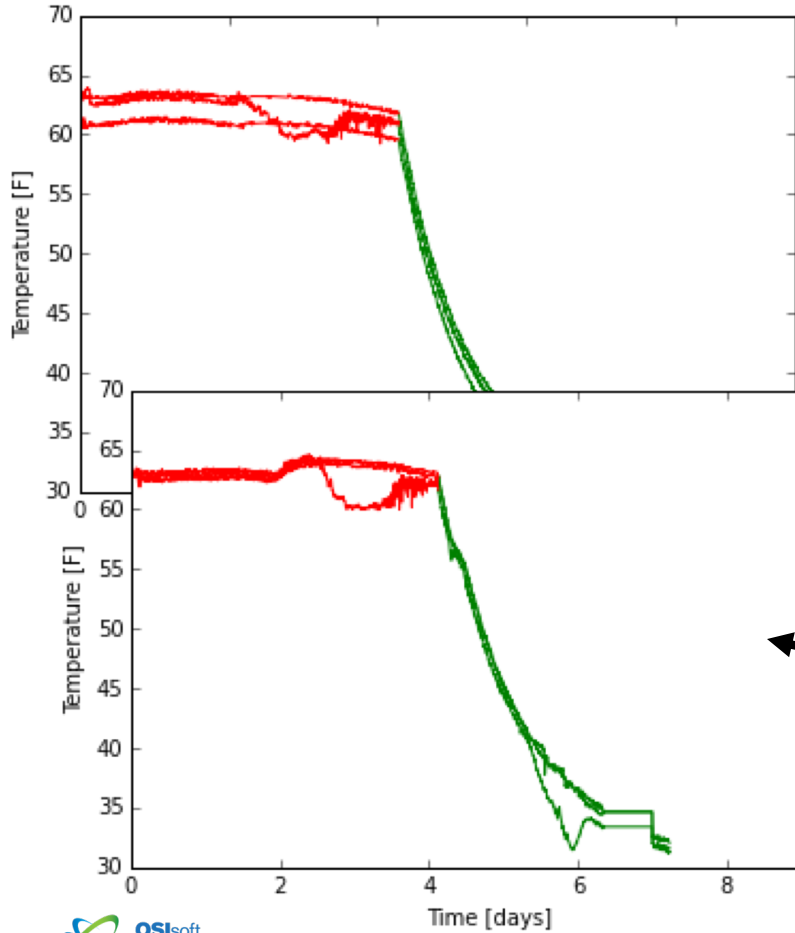


```
# Predict PCs
beers.projected <- predict(beers.pca,
  newdata=beers.data)

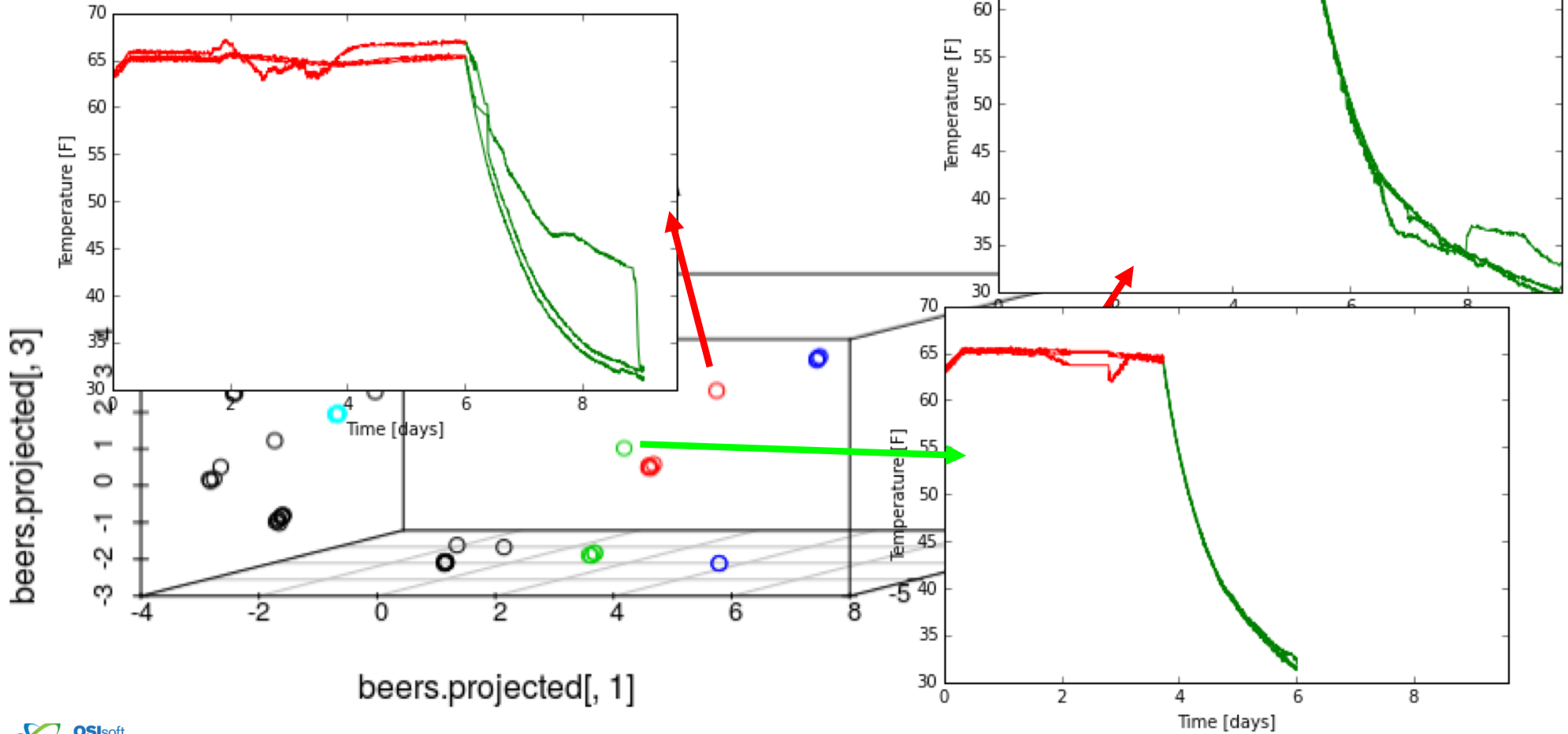
scatterplot3d(beers.projected[,1],
  beers.projected[,2],
  beers.projected[,3],
  color = beers.labels,
  main = "PCA")
```

- "Realtime Hops"
- "Red Wonder"
- "Alistair"
- "Kennedy"
- "Kerberos"

Similar members



Outliers



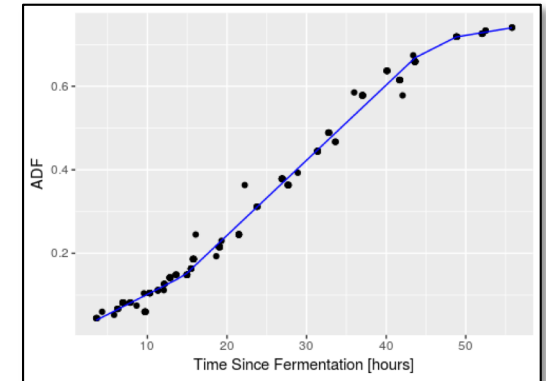


Lessons Learned

Lesson Learned



Predicting Apparent Degree of Fermentation (ADF)



CHALLENGE

Linear model of ADF as a function of time is insufficient

- ADF starts negative
- Errors are correlated with predicted values (BAD!)
- Normality test values are not near model (not presented)

SOLUTION

Use domain knowledge to develop a better model

- Existence of three phases of fermentation suggests three segments in the linear model

RESULTS

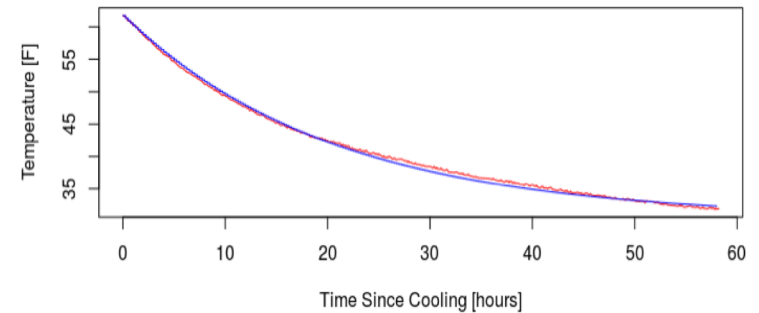
Piecewise linear model is much improved over basic linear model

- Errors are not correlated with predicted values
- Normality test values are better (not presented)
- Could consider other attributes

Lesson Learned



Predict cooling of fermented beer



CHALLENGE

Want to predict cooling trend, completion times

- Could build complex model
- Many attributes available, including starting temperature, volume, and time

SOLUTION

Basic thermodynamics function for cooling

- Use data to find values of constants

RESULTS

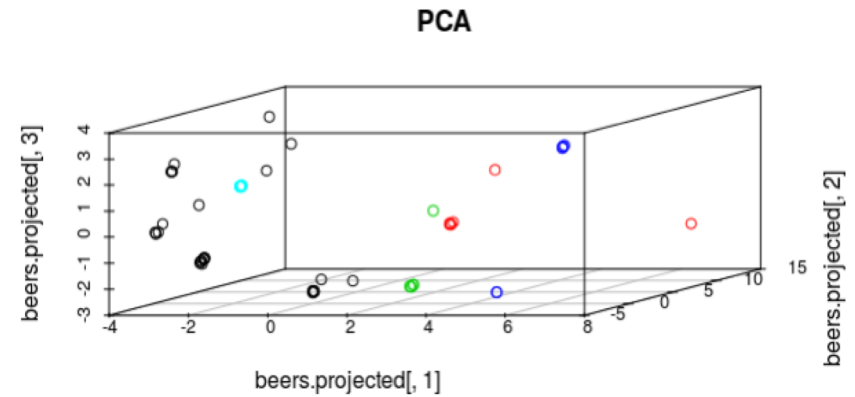
Basic cooling model works well

- Errors are not correlated with predicted values

Lesson Learned



Visualizing high-dimensional data



CHALLENGE

Want to see similarities and differences between runs

- Data in form of time series with many attributes
- Humans need at most 3-D data

SOLUTION

Use PCA on run-focused data

- Represent situation of interest
- Identify reduced dimensionality

RESULTS

Can see outliers and patterns

- Some beers are consistent
- Others include outliers

References for further study

- Gebhard Kirchgässner and Jürgen Wolters, *Introduction to Modern Time Series Analysis*, Springer Berlin Heidelberg New York, 2007.
- Peter J. Brockwell and Richard A. Davis, *Introduction to Time Series and Forecasting*, 2nd ed., Springer Texts in Statistics, Springer Science+Business Media, LLC, New York, NY, USA, 2002.
- James D. Hamilton, *Time Series Analysis*, Princeton University Press, Princeton, NJ, USA, 1994.
- NIST/SEMATECH e-Handbook of Statistical Methods, <http://www.itl.nist.gov/div898/handbook/>, 2018.
- <http://www.r-tutor.com>
- http://wineserver.ucdavis.edu/industry/enology/fermentation_management/wine/problem_fermentations.html

Contact Information



Brian D. Davison

davison@cse.lehigh.edu

Associate Prof., Lehigh University

Slides and R source code available from:

<http://www.cse.lehigh.edu/~brian/PIW18/>



Lehigh University's Linderman Library