Lecture 3: Hat’s off!

Last time

We introduced a data set that we will use to illustrate some of the basic concepts behind regression diagnostics -- It is just big enough to be non-trivial (barely) and exhibits non-linear structure...

PNAS PNAS PNAS

Estimating least-developed countries' vulnerability to climate-related extreme events over the next 50 years

Anthony G. Patt[a,1], Mark Tadross[b], Patrick Nussbaumer[c], Kwabena Asante[d], Marc Metzger[e,f], Jose Rafael[g], Anne Goujon[a,h], and Geoff Brundrit[i]

[a]International Institute for Applied Systems Analysis, 2361 Laxenburg, Austria; [b]Climate Systems Analysis Group, University of Cape Town, Rondebosch 7701, South Africa; [c]Institute of Environmental Science and Technology, Autonomous University of Barcelona, 08193 Bellaterra, Spain; [d]Climatus LLC, Mountain View, CA 94041; [e]Centre for the Study of Environmental Change and Sustainability, University of Edinburgh, EH8 9XP, Scotland; [f]Alterra, Wageningen University and Research Centre, 6700 AA Wageningen, The Netherlands; [g]Department of Geography, University of Eduardo Mondlane, Maputo, Mozambique; [h]Vienna Institute of Demography, Austrian Academy of Sciences, 1040 Vienna, Austria; and [i]Department of Oceanography, University of Cape Town, Rondebosch 7701, South Africa

Edited by Stephen H. Schneider, Stanford University, Stanford, CA, and approved December 4, 2009 (received for review September 10, 2009)

When will least developed countries be most vulnerable to climate change, given the influence of projected socio-economic development? The question is important, not least because current levels of international assistance to support adaptation lag more than an order of magnitude below what analysts estimate to be needed, and scaling up support could take many years. In this paper, we examine this question using an empirically derived model of human losses to climate-related extreme events, as an indicator of vulnerability and the need for adaptation assistance. We develop a set of 50-year scenarios for these losses in one country, Mozambique, using high-resolution climate projections, and then extend the results to a sample of 23 least-developed countries. Our approach takes into account both potential changes in countries' exposure to climatic extreme events, and socio-economic development trends that influence countries' own adaptive capacities. Our results suggest that the effects of socio-economic development trends may

sensitivity to those stressors, which in turn is determined by a complex set of social, economic, and institutional factors collectively described as determining its adaptive capacity (5, 6). As the UNFCCC secretariat suggested in its needs assessment, "one of the key limitations in estimating the costs of adaptation is the uncertainty about adaptive capacity. Adaptive capacity is essentially the ability to adapt to stresses such as climate change. It does not predict what adaptations will happen, but gives an indication of differing capacities of societies to adapt *on their own* to climate change or other stresses" (1, p. 97).

Human losses to extreme weather events can serve as a reliable indicator for this vulnerability, and with it the need for financial assistance, for two reasons. First, measures to reduce vulnerability to extreme weather events account for a particularly large share of estimated adaptation financial needs (1). Second, in the context of efforts to achieve a wide range of development goals, it is only

Last time

We introduced a data set that we will use to illustrate some of the basic concepts behind regression diagnostics -- It is just big enough to be non-trivial (barely) and exhibits non-linear structure

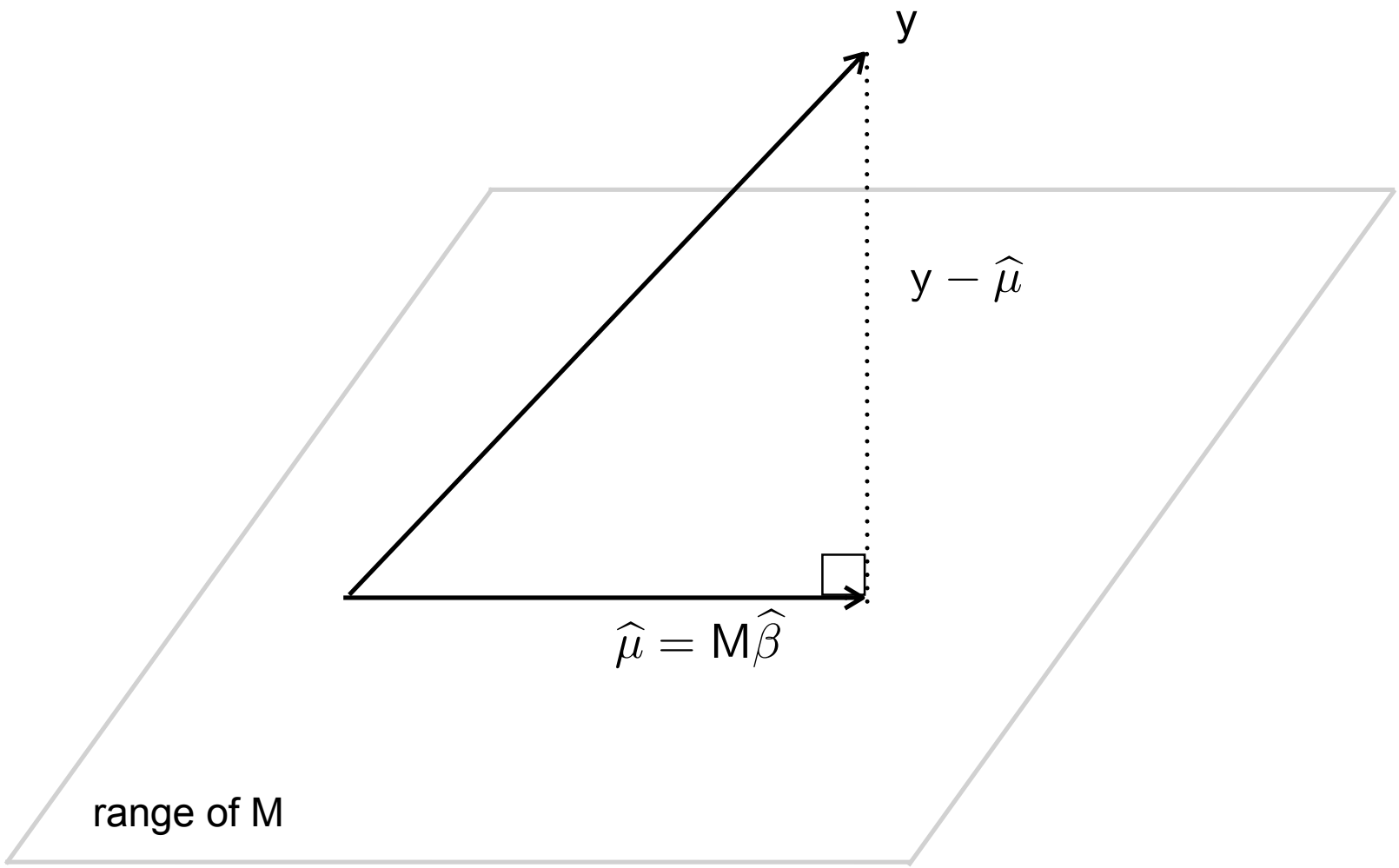
We then reviewed a handful of results from linear algebra -- Just enough to help us interpret regression as a kind of projection, complete with a Pythagorean theorem (we will see this kind of relationship for generalized linear models as well, but our loss function will be Kullback-Leibler divergence)

A geometric view

This also means that we have partitioned the length of the vector y into two components, one for the (estimated) systematic part and one for the (estimated) random component

$$\begin{aligned} \|\mathsf{y}\|^2 &= \|\mathsf{y} - \widehat{\mu} + \widehat{\mu}\|^2 \\ &= \|\mathsf{y} - \widehat{\mu}\|^2 + \|\widehat{\mu}\|^2 + 2\widehat{\mu}^{\mathsf{t}}(\mathsf{y} - \widehat{\mu}) \\ &= \|\mathsf{y} - \widehat{\mu}\|^2 + \|\widehat{\mu}\|^2 \end{aligned}$$

This is essentially the Pythagorean theorem and we can see it graphically on the following (somewhat lame) slide

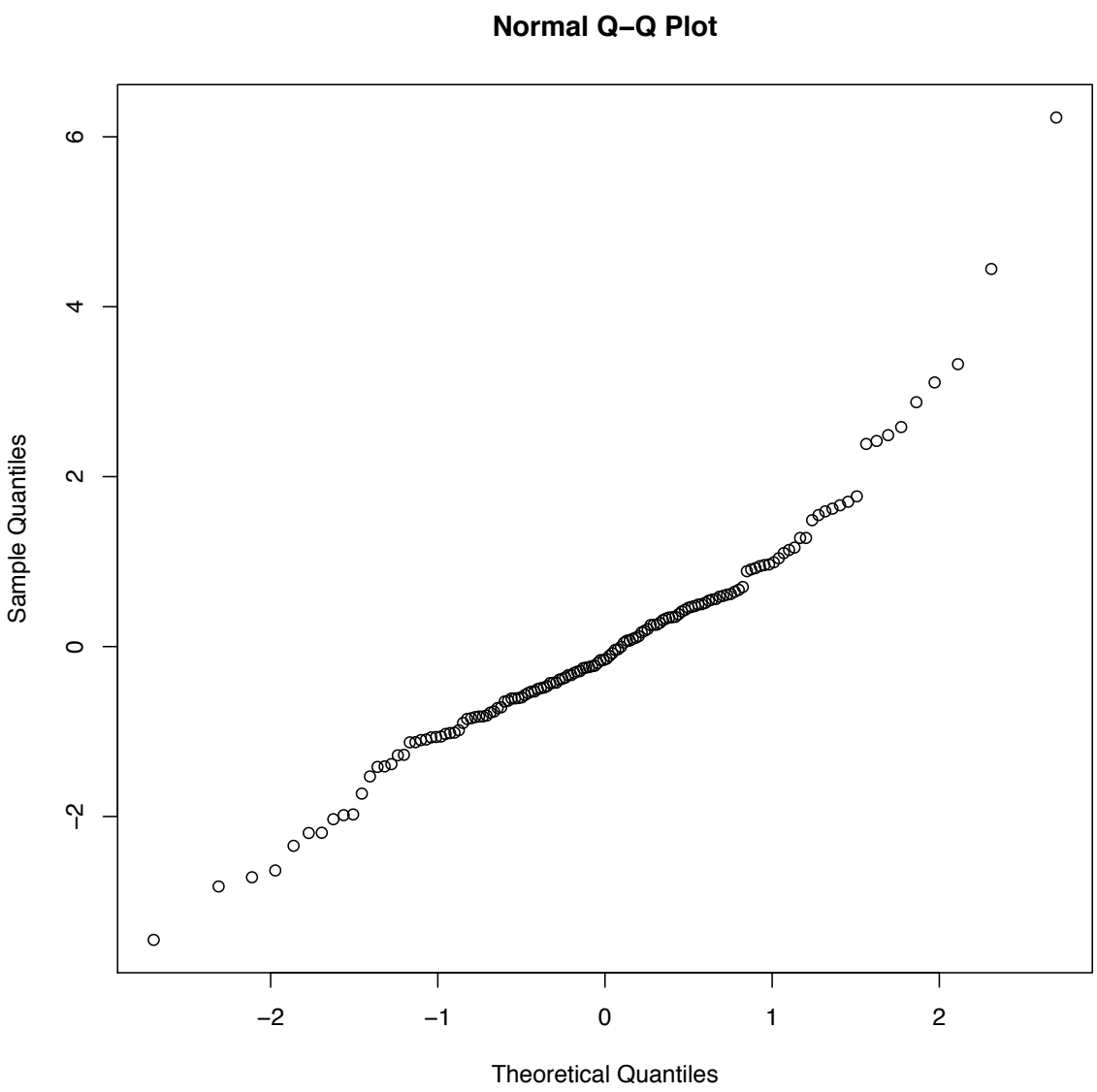

y
$y - \widehat{\mu}$
$\widehat{\mu} = M\widehat{\beta}$
range of M

Last time

We introduced a data set that we will use to illustrate some of the basic concepts behind regression diagnostics -- It is just big enough to be non-trivial (barely) and exhibits non-linear structure

We then reviewed a handful of results from linear algebra -- Just enough to help us interpret regression as a kind of projection, complete with a Pythagorean theorem (we will see this kind of relationship for generalized linear models as well, but our loss function will be Kullback-Leibler divergence)

And we had a brief review of Q-Q plots...


Normal Q–Q Plot
Sample Quantiles
Theoretical Quantiles
6
4
2
0
−2
−2
−1
0
1
2

Today

We are going to start with the hat matrix, picking up where we left off and hopefully making our last few minutes a bit more understandable -- We will also introduce two other ways to think about the concept of “leverage”

Next, we’ll return to the vulnerability data and examine the best way to include polynomial terms -- Along the way, we will see another set of regression diagnostics that describe the dependence structure among our predictor variables

If there is time, we will examine successive orthogonalization via the Gram-Schmidt method -- A key tool in deriving one of the important (stable) decompositions used in actually fitting a least squares regression

From last time

We'll start with the hat matrix H -- It takes us from our observed data to an estimate of the vector of conditional means

$$\widehat{\mu}_j = \sum_{i=1}^{n} h_{ji} y_i$$

if we let

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1n} \\ h_{21} & h_{22} & \cdots & h_{2n} \\ \vdots & \vdots & & \vdots \\ h_{n1} & h_{n2} & \cdots & h_{nn} \end{bmatrix}$$

The hat matrix

Given this expression, we see that the elements h_{ij} express the degree of "leverage" that an observation y_i has on a fitted value $\widehat{\mu}_j$

In general, the greatest impact that y_i has on the fit is through $\widehat{\mu}_i$, and hence people often focus their attention on the diagonal elements h_{ii} of the hat matrix

Because H is symmetric and idempotent, we can write

$$h_{ii} = \sum_j h_{ij}^2 = h_{ii}^2 + \sum_{j \neq i} h_{ij}^2 \geq 0$$

Therefore we see that the diagonal elements satisfy $0 \leq h_{ii} \leq 1$

The hat matrix

The bound helps, but it's hard to know when a particular value is "big" -- We can show that the eigenvalues of a projection matrix are either 0 or 1

This sounds bad but it's pretty easy to derive once we have an orthogonal basis for the column space of M -- We'll do this later in today's class when we look at Gram-Schmit

If $q_1, \ldots, q_p$ form an orthonormal basis for the columnspace of M, and we let $q_{p+1}, \ldots, q_n$ denote an orthonormal basis for the orthogonal complement of this space, we can create a new matrix $Q = [q_1, \ldots, q_n]$

The hat matrix

Then multiplying by H gives (remember $q_1, \ldots, q_p$ will project onto themselves)

$$HQ = H\left[q_1, \ldots, q_p, q_{p+1}, \ldots, q_n\right] = \left[q_1, \ldots, q_p, 0, \ldots, 0\right]$$

so that $Q^t HQ = \mathrm{diag}(1, \ldots, 1, 0, \ldots, 0)$, with p one's

The hat matrix

So, the eigenvalues of the hat matrix are either 0 or 1 and the total number of 1's matches the rank of M -- that means that the trace of H (the sum of its diagonal elements) which is also the sum of its eigenvalues is just p

Long story short, we have

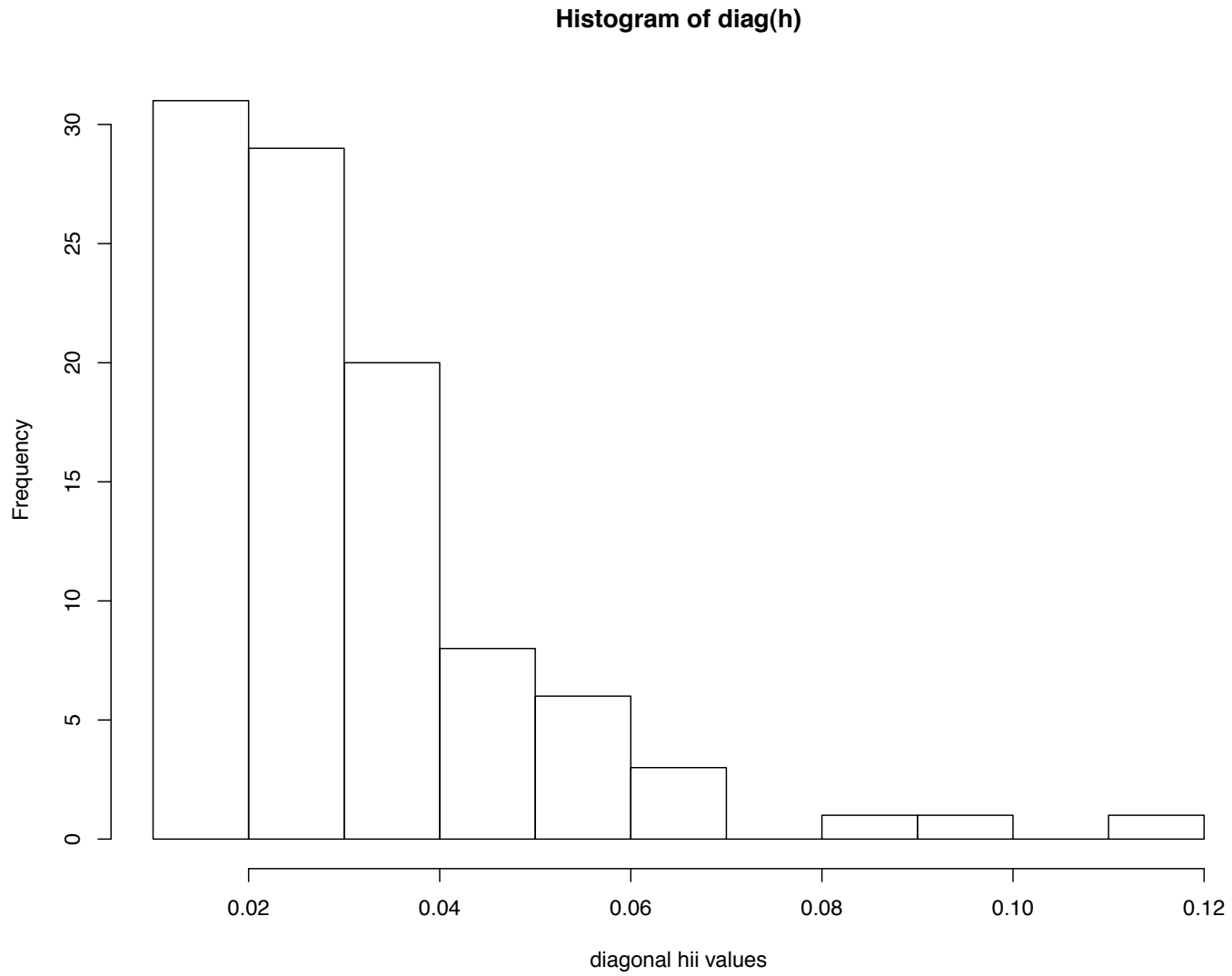
$$\sum_{i=1}^{n} h_{ii} = p \quad \text{or} \quad \frac{1}{n}\sum_{i=1}^{n} h_{ii} = p/n$$

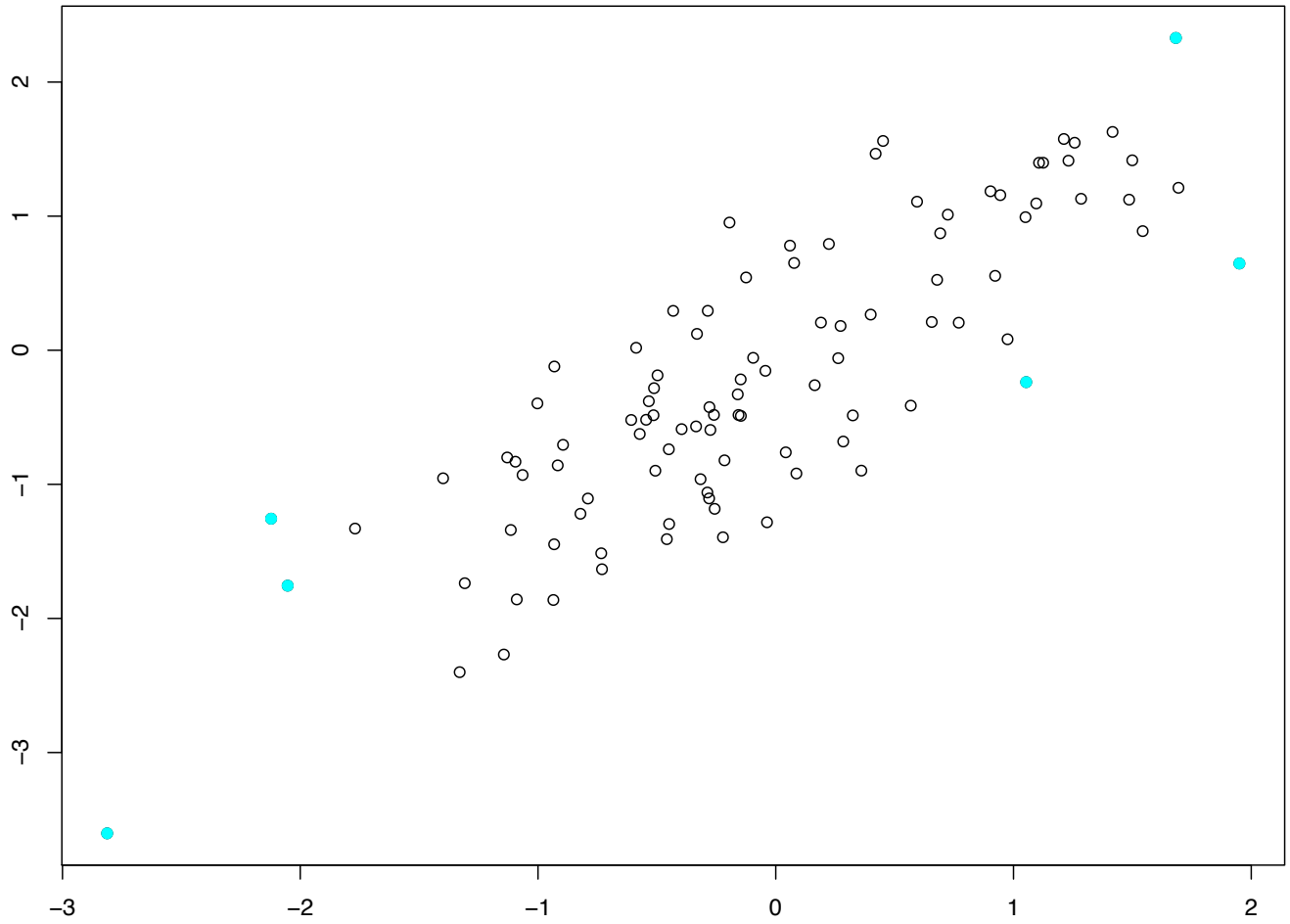
The hat matrix

In searching for high-leverage observations, then, we often look for “big” values along the diagonal of the hat matrix, where big is larger than 2p/n

On the next slide, we have simulated predictors (bivariate normal) -- 100 rows and p=3 (an intercept and two predictors) translates to a “cutoff” of 0.06

The histogram shows the 100 leverage values, six of which are above the cutoff... the points are colored cyan on the following plot -- Do they match your intuition?

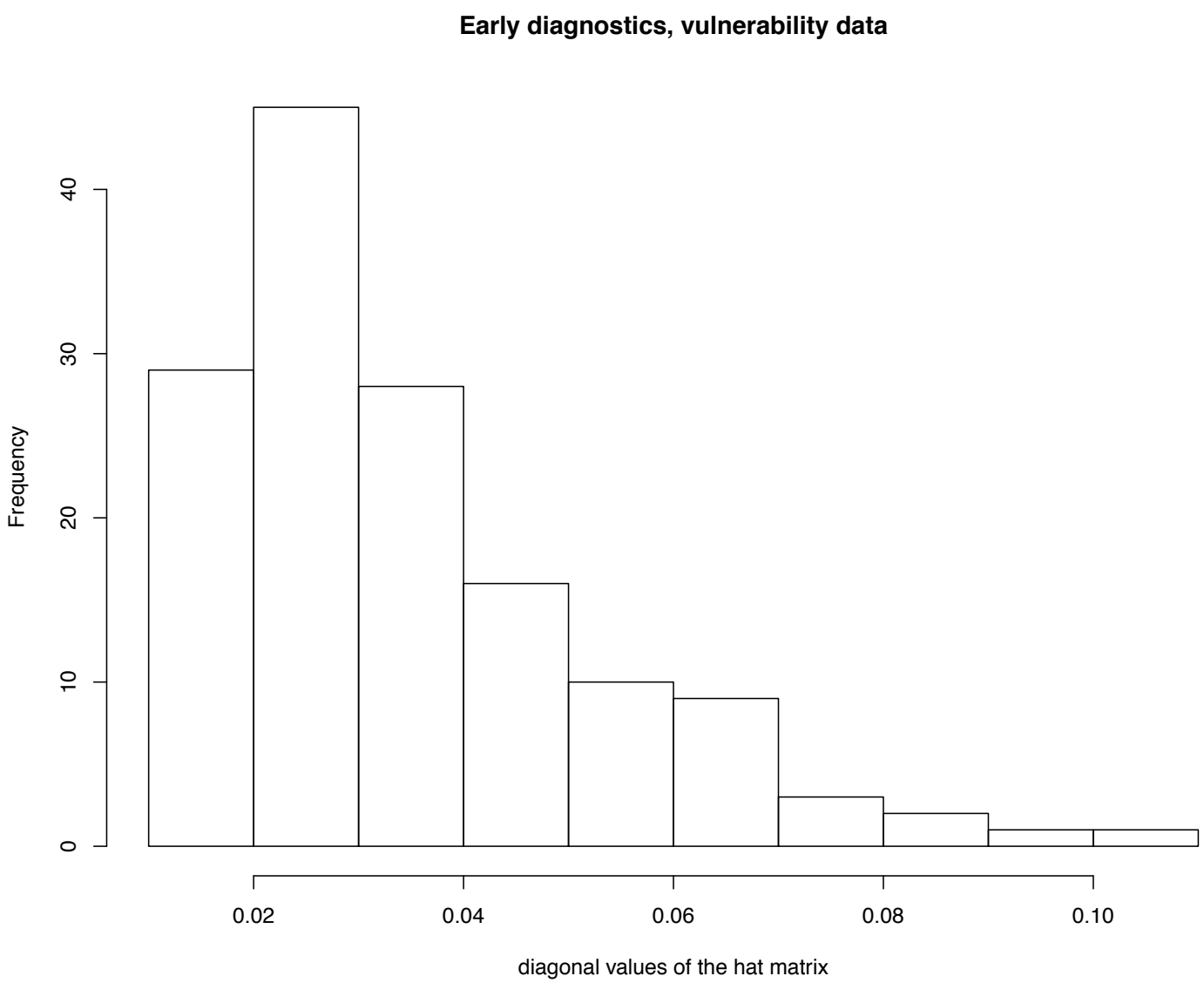

Histogram of diag(h)
Frequency
0
5
10
15
20
25
30
0.02
0.04
0.06
0.08
0.10
0.12
diagonal hii values

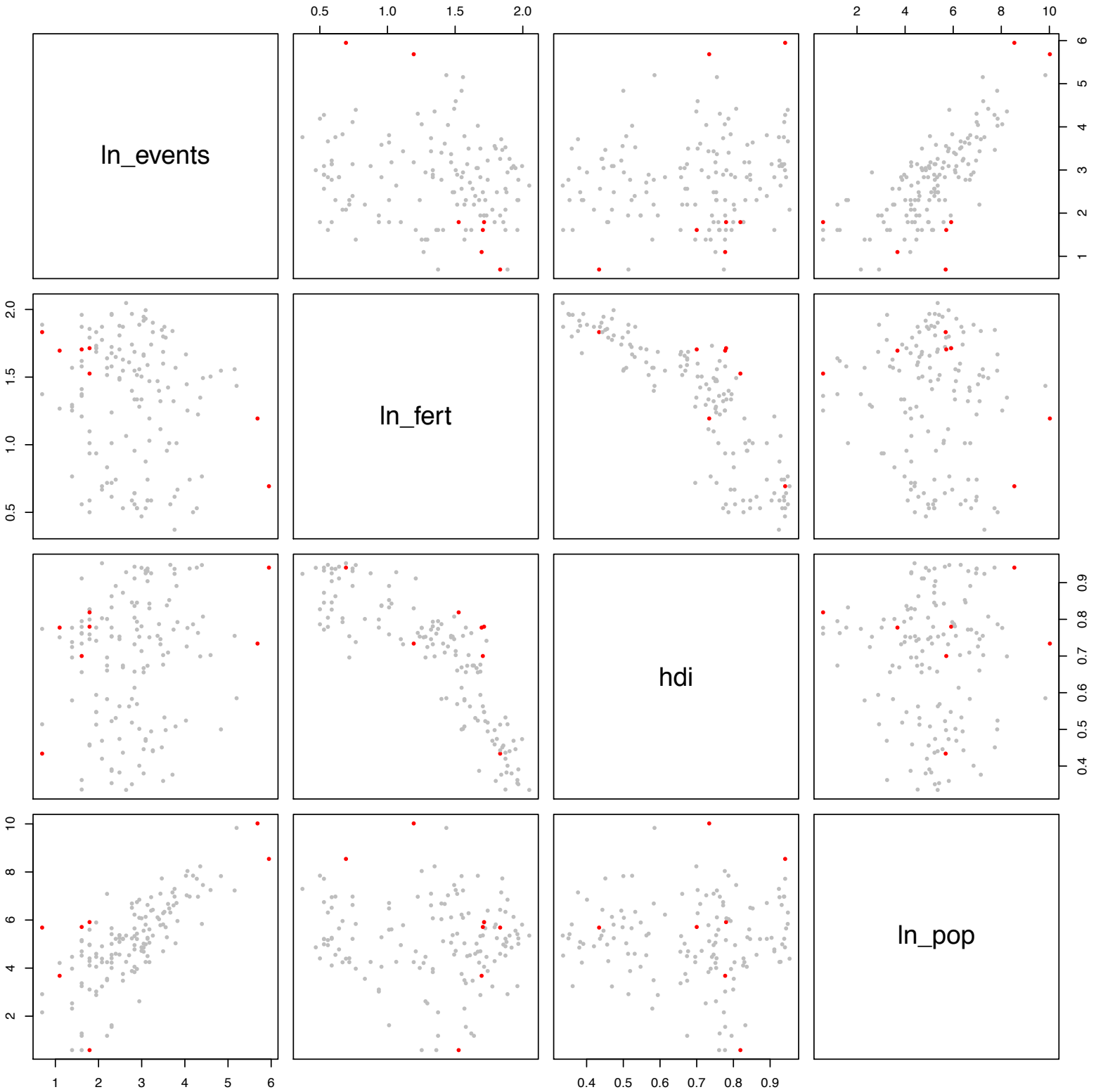

2
1
0
−1
−2
−3
−3
−2
−1
0
1
2

Applying the metric

Applying this to our simple model, we find that there are a handful of points that seem to have “big” leverage on the fits -- In R we can easily compute the hat matrix values and have a look

```
> inf = lm.influence(fit)
> names(inf)
[1] "hat"          "coefficients" "sigma"        "wt.res"

# the cutoff
> 10/nrow(vul)
[1] 0.06944444

> vul[inf$hat>0.069,1]
[1] China P Rep     Cote d'Ivoire   Oman            Saudi Arabia
[5] Syrian Arab Rep Tonga           United States
144 Levels: Albania Algeria Angola Argentina Armenia Australia ... Zimbabwe
```


Early diagnostics, vulnerability data
Frequency
0
10
20
30
40
0.02
0.04
0.06
0.08
0.10
diagonal values of the hat matrix


ln_events
ln_fert
hdi
ln_pop
0.5
1.0
1.5
2.0
2
4
6
8
10
1
2
3
4
5
6
0.5
1.0
1.5
2.0
0.4
0.5
0.6
0.7
0.8
0.9
2
4
6
8
10
1
2
3
4
5
6
0.4
0.5
0.6
0.7
0.8
0.9

The hat matrix

The goal of going down this path is simply to show that certain constructions from linear algebra have interpretations in statistical terms -- The elements of a projection matrix providing us insight into points that are exerting undo “influence” on the fit

Using the hat matrix we can also assess the effect of removing, say, a single point -- Using the Sherman-Morrison-Woodbury formula, for example, we can derive the effect on the variance-covariance matrix by dropping a point

Homework (finally)

It will also turn out that certain elements of H are extremely useful in understanding alternations to our data -- Say, for example, we want to leave a point out

To judge where we are in terms of our familiarity with linear algebra, let's take a first big step in deriving a class of regression diagnostics -- We will prove the so-called Sherman-Morrison-Woodbury formula

Let A be an invertible square (pxp) matrix and let u and v be two columnb vectors, each of length p. Establish the following equivalence:

$$(A + uv^t)^{-1} = A^{-1} - \frac{A^{-1}uv^tA^{-1}}{1 + v^tA^{-1}u}$$

Dropping a point

Let m_i denote the ith row of the matrix M and let $M_{(i)}$ denote the (n-1)-by-p matrix with this row removed -- We can show that the new variance-covariance matrix associated with this reduced fit is

$$(M_{(i)}^t M_{(i)})^{-1} = (M^t M)^{-1} + \frac{(M^t M)^{-1} m_i^t m_i (M^t M)^{-1}}{1 - h_{ii}}$$

which means removing a point with a high value of h_{ii} can increase the variance of our coefficient estimates -- Does this make sense?

Another interpretation: Equivalent number of observations

Let's again cement notation -- We are going to assume we have n paired observations (inputs and responses, independent and dependent variables) that we denote $(x_{i1}, \ldots, x_{ip})$, y_i, for $i = 1, \ldots, n$

The ordinary least squares fit to these data, the coefficients that minimize

$$\sum_{i=1}^{n} (y_i - \beta_1 x_{i1} - \beta_p x_{ip})^2$$

can be written as $\widehat{\beta} = (M^t M)^{-1} M^t y$ with the estimated conditional means expressed as the product $\widehat{\mu} = M(M^t M)^{-1} M^t y = Hy$, where M is the model matrix associated with the n predictors

Equivalent number of observations

First note that we can rewrite the residuals from an OLS fit in terms of the hat matrix elements -- For convenience we'll let h_i denote the ith diagonal element h_{ii}

$$\widehat{\epsilon}_i = y_i - \widehat{\mu}_i = (1 - h_i)y_i - \sum_{j \neq i} h_j \, y_j$$

From here we see that if h_i is close to 1 "a gross error in y_i will not necessarily show up in $\widehat{\epsilon}_i$ " -- In this sense, we say that data points associated with large values of h_i have high leverage

Huber (1981) describes $1/h_i$ as the equivalent number of observations entering into the determination of $\widehat{\mu}_i$ -- So if the diagonal element h_i is 1, then the point is determined solely by one point, namely y_i

Huber suggests that $h_i > 0.5$ (2 equivalent observations) is clearly large but that $h_i > 0.2$ (5 equivalent observations warrants special attention)

Equivalent number of observations

As we have seen, calibrating diagnostics is important -- We would like to **establish a scale by which we judge when something is large** (in this case)

In some cases **we use probabilistic arguments** (number of standard deviations from the mean, say) -- Here Huber examines the hat matrix elements in terms of **an equivalent number of observations**

Over the course of the quarter, we'll see common scales related to equivalent observations or, to measure the amount of effort going into a fit, equivalent degrees of freedom -- Basically, **taking something we can relate to statistically and cast it as a scale for some diagnostic**

Equivalent number of observations

To make this intuition precise, assume that $h_i = 1/k$ for some value of k, then if we duplicate the ith row of M then h_i is changed to 1/(k+1) -- So if h_i is big, it can be decreased by duplication or triplication of the observation at the ith input point

To establish this, we can again appeal to a simplification of the Sherman-Morrison-Woodbury formula -- If M is our original n-by-p model matrix, let M^+ denote the new model matrix obtained by adding a row

$$M^+ = \begin{bmatrix} M \\ m^t \end{bmatrix}$$

where m is a p-by-1 column vector representing a new input data point

Equivalent number of observations

Now, assume M is orthogonal, so that $M^t M = I_{p\times p}$ and we can write

$$(M^+)^t M^+ = I_{p\times p} + mm^t$$

Then, by a simple application of the Sherman-Morrison-Woodbury formula

$$\left[(M^+)^t M^+\right]^{-1} = I_{p\times p} - \frac{mm^t}{1+m^t m}$$

And from here, the new hat matrix $H^+ = M^+\left[(M^+)^t M^+\right]^{-1}(M^+)^t$ is just

$$H^+ = \left[\begin{array}{c} M \\ \hline m^t \end{array}\right]\left(I - \frac{mm^t}{1+m^t m}\right)\left[\begin{array}{c|c} M^t & m \end{array}\right]$$

$$= \left[\begin{array}{c|c} MM^t - \frac{(Mm)(Mm)^t}{1+m^t m} & \frac{Mm}{1+m^t m} \\ \hline \frac{(Mm)^t}{1+m^t m} & \frac{m^t m}{1+m^t m} \end{array}\right]$$

Equivalent number of observations

Again, assuming we have have orthogonal predictors, then our original hat matrix is $H = MM^t$

Therefore, if we duplicate say the nth row, letting m^t be the nth row of M, then we have that $h_n = m^t m$ and so the (n+1)st diagonal element of H^+ is

$$h_{n+1} = \frac{h_n}{1 + h_n}$$

where we have dropped the tilde over h_{n+1} as it unambiguously refers to the updated hat matrix H^+

Therefore, if h_n is 1/k, h_{n+1} is 1/(1+k) -- Because the diagonal elements of the hat matrix scale in this way when you add duplicate observations, Huber interprets these values like an equivalent number of observations going into the estimation of a conditional mean value

Another interpretation: Mahalanobis distance

Throughout this term, we'll depend on various notions of distance between points (or between points and model predictions or between models) as the basis for statistical methodology -- Heck, least squares is just Euclidean distance

For ease of notation, again consider a new data point m that we will add to an existing set of observations, but this time we'll try to understand the hat matrix in terms of a distance

Mahalanobis Distance

Suppose that we have “centered” each predictor variable -- That is, we have subtracted the mean off each so that

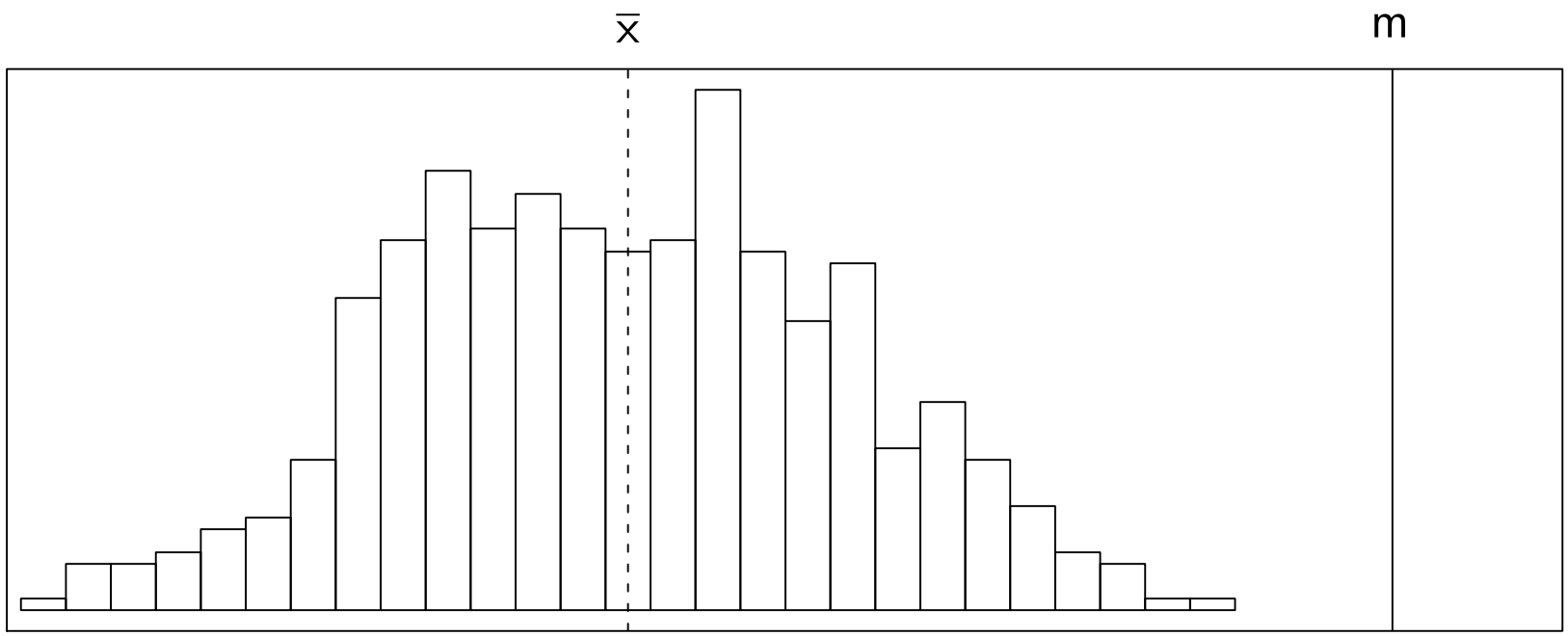
$$\sum_{i=1}^{n} x_{ij} = n\bar{x}_j = 0 \quad \text{for} \quad j = 1, \ldots, p$$

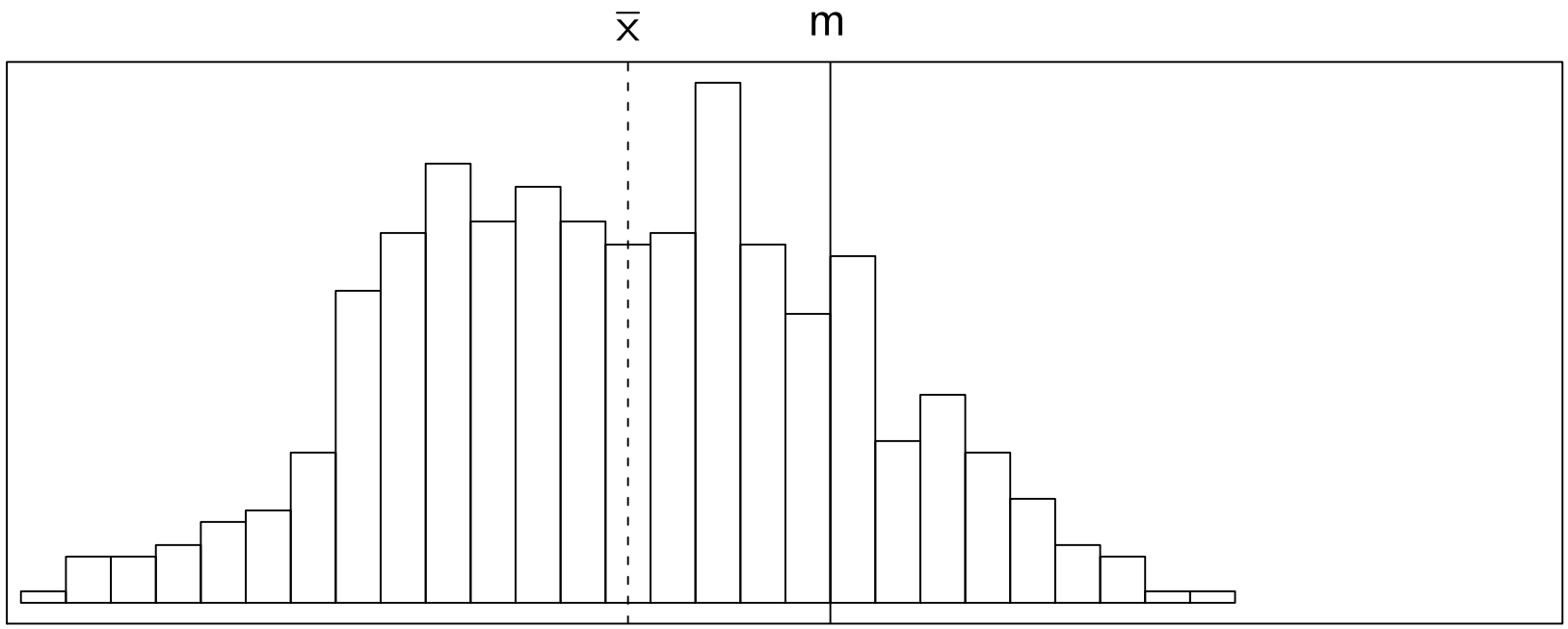
Equivalently, we can think of this as the first step in Gram-Schmidt, where we have regressed out the constant vector from each of our predictors -- That is, we’re now modeling with the residuals of these regressions

Mahalanobis Distance

Suppose for the moment we have just a 1-dimensional predictor (p=1) so that our new point m is just a scalar value -- How might we judge the distance between this point and the other $x_1, \ldots, x_n \in \mathbb{R}$?

Well, the simple Euclidean distance (or any of the Holder norms for that matter) could be use so that we might consider $|m - \bar{x}|$, for example -- But as statisticians, is that all there is?


$\bar{x}$
m


$\bar{x}$
m

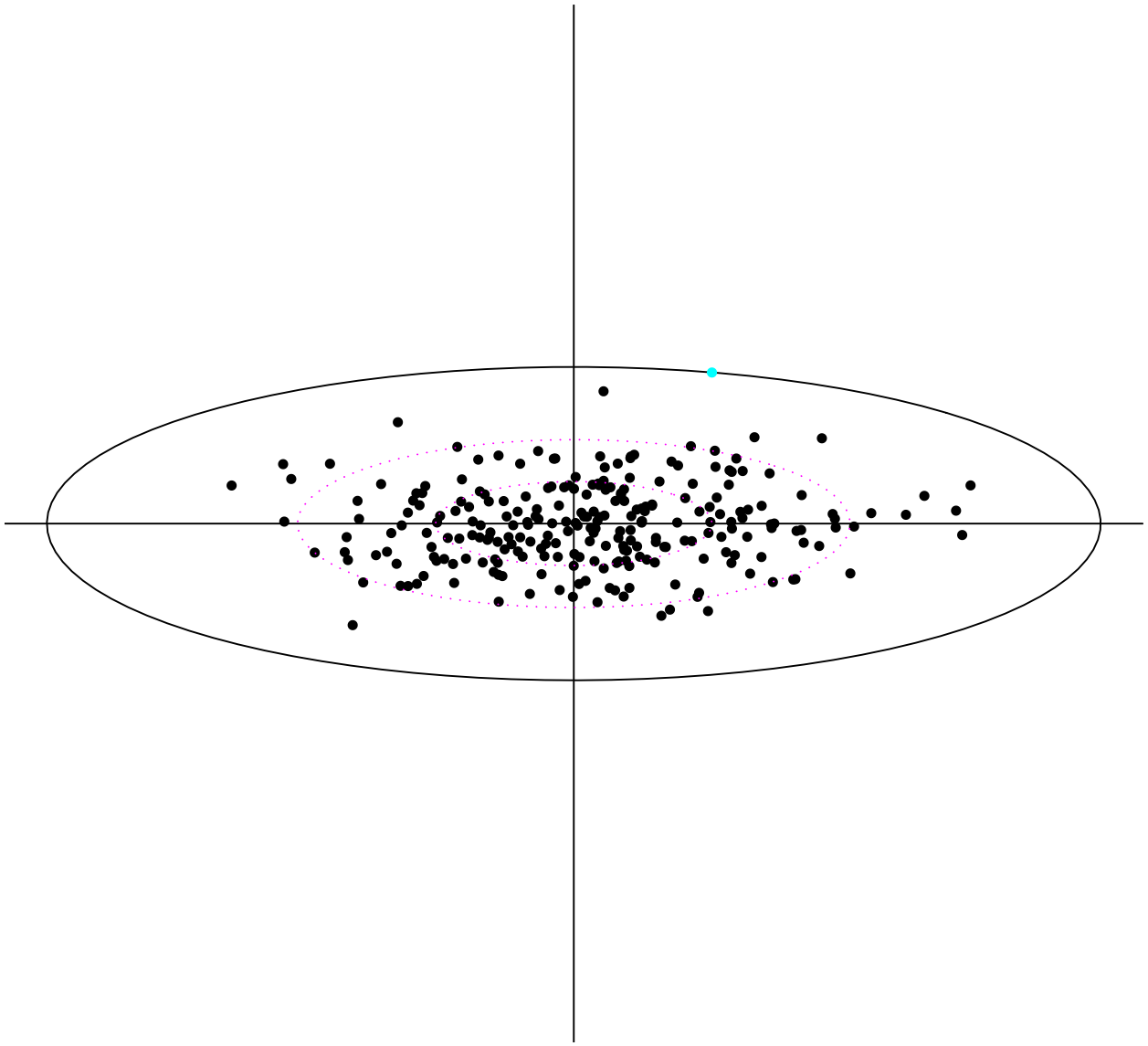
Mahalanobis Distance

In these cases, we would probably try to consider using the standard deviation as a scale for interpreting what's close by and what's far away -- In the first case, we are about 3.5 standard deviations from the "center" of the data (the mean) and in the second, we're just 1 standard deviation away

Mahalanobis distance, then, is an attempt to generalize this idea to higher-dimensional data -- From univariate observations to vectors

To get there, suppose our predictors are observations (x_{i1}, x_{i2}) for i=1,...,n, from two independent (stochastically) normal distributions with variances, then instead of the Euclidean distance between a new point $m = (m_1, m_2)$ and the "center" $(\bar{x}_1, \bar{x}_2)$, we would instead weight the distances

$$\sqrt{\frac{(m_1 - \bar{x}_1)^2}{s_1^2} + \frac{(m_2 - \bar{x}_2)^2}{s_2^2}}$$



```
# tedious but at least you see how to draw ellipses

x1 <- rnorm(250)
x2 <- rnorm(250)/3

m <- c(1,1.2)
d <- sqrt((m[1]-mean(x1))^2/var(x1) + (m[2]-mean(x2))^2/var(x2))

library(ellipse)
p = ellipse(matrix(c(var(x1),0,0,var(x2)),ncol=2),t=d)
plot(p,type="l",xlim=range(p),ylim=range(p),axes=F,xlab="",ylab="",pty="s")
points(x1,x2,pch=20)
abline(v=0)
abline(h=0)
points(m[1],m[2],col=5,pch=20)

p = ellipse(matrix(c(1,0,0,(1/3)^2),ncol=2),t=2,col=5)
lines(p,lty=3,col=6)
p = ellipse(matrix(c(1,0,0,(1/3)^2),ncol=2),t=1,col=5)
lines(p,lty=3,col=6)
```

Mahalanobis Distance

In this case, we measure the distance between a new point (in cyan) and the data set using a weighted least squares criterion -- The pink dashed ellipses we draw correspond to t-statistics with the value 1 and 2 while the outer solid curve corresponds to the weighted distance between m=(1,1.2) and the center of the data

We can think of this change as moving from a standard (squared) Euclidean distance given by

$$d^2(m, \bar{x}) = (m_1 - \bar{x}_1)^2 + (m_2 - \bar{x}_2)^2 = (m - \bar{x})^t (m - \bar{x})$$

to a weighted norm given by

$$d^2(m, \bar{x}) = \frac{(m_1 - \bar{x}_1)^2}{\text{var}\, x_1} + \frac{(m_2 - \bar{x}_2)^2}{\text{var}\, x_2} = (m - \bar{x})^t \begin{bmatrix} \text{var}\, x_1 & 0 \\ 0 & \text{var}\, x_2 \end{bmatrix}^{-1} (m - \bar{x})$$

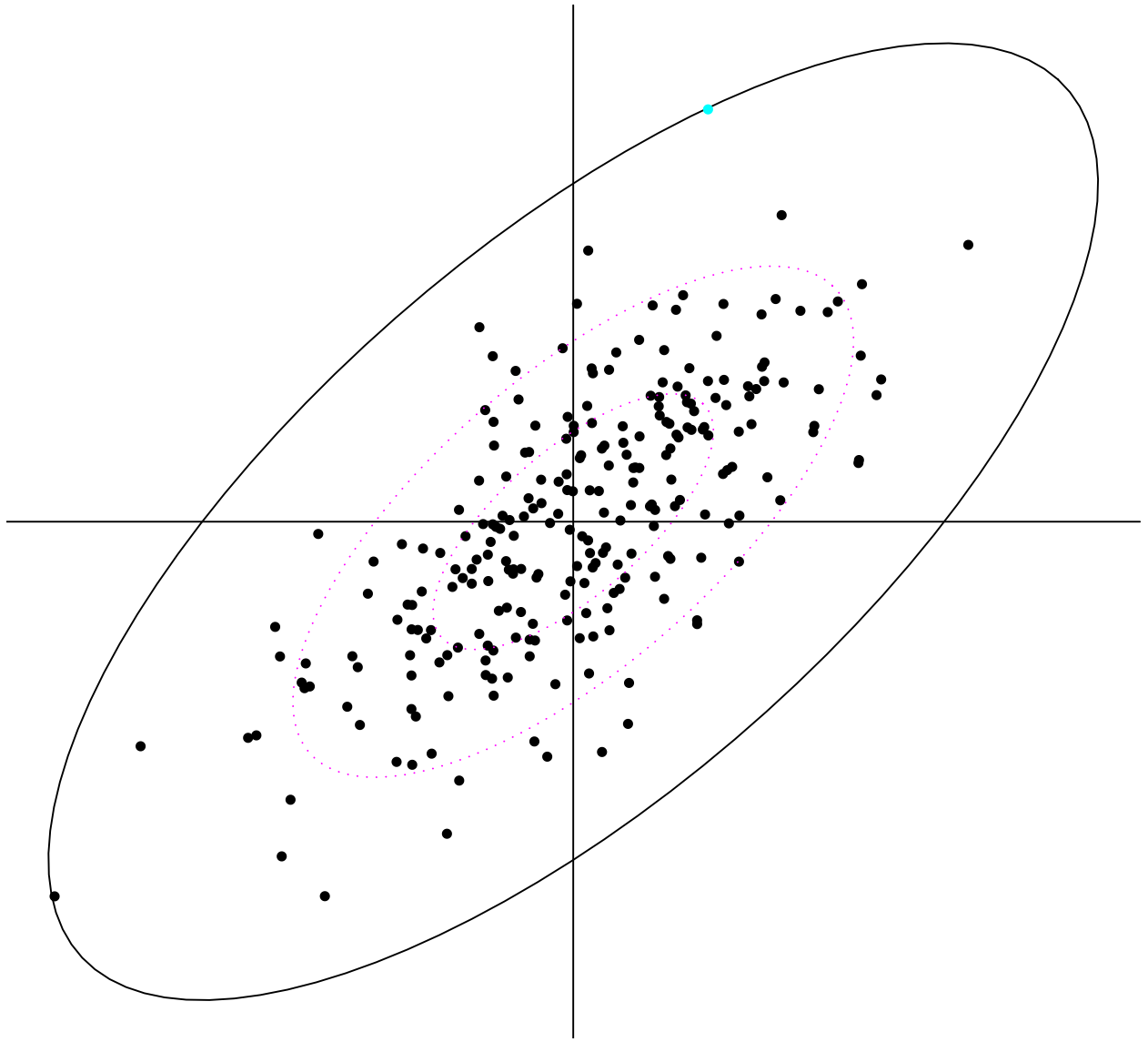
Mahalanobis Distance

In effect, what we are doing is using information about the distribution of the input variables to provide us with a stochastic notion of what's nearby and what's far away -- It is directly analogous to the t-statistic we started with

So finally, if we assume that our two inputs are observations from a general bivariate normal distribution, we take the Mahalanobis distance of a new point m to the center of the data $(\bar{x}_1, \bar{x}_2)$ to be

$$d^2(m, \bar{x}) = (m - \bar{x})^t \Sigma^{-1} (m - \bar{x}) = (m - \bar{x})^t \begin{bmatrix} \operatorname{var} x_1 & \operatorname{cov}(x_1, x_2) \\ \operatorname{cov}(x_1, x_2) & \operatorname{var} x_2 \end{bmatrix}^{-1} (m - \bar{x})$$

where Σ is just the (estimated) variance-covariance matrix for our inputs



```
# tedious but at least you see how to draw ellipses

x1 <- rnorm(250)
x2 <- (x1 + rnorm(250))/6

m <- c(1,0.75)
Sig <- matrix(c(var(x1),cov(x1,x2),cov(x1,x2),var(x2)),ncol=2)
d <- sqrt((m-c(mean(x1),mean(x2)))%*%solve(Sig)%*%(m-c(mean(x1),mean(x2))))

library(ellipse)
p = ellipse(Sig,t=d)
plot(p,type="l",axes=F,xlab="",ylab="",pty="s")
points(x1,x2,pch=20)
abline(v=0)
abline(h=0)
points(m[1],m[2],col=5,pch=20)

p = ellipse(Sig,t=2,col=5)
lines(p,lty=3,col=6)
p = ellipse(Sig,t=1,col=5)
lines(p,lty=3,col=6)
```

Mahalanobis Distance

So you will probably recognize this form from the expression for a multivariate normal distribution -- You can also get a sense of what this is doing by examining the eigen-structure of Σ

Recall that any symmetric p-by-p matrix A has a spectral decomposition (an eigen-decomposition) of the form

$$A = O\,D\,O^t$$

where O is a p-by-p matrix with orthonormal columns so that $O^tO = I_{p\times p}$ or $O^{-1} = O^t$, and D is a p-by-p diagonal matrix (of real values)

Mahalanobis Distance

Now, let $X = (X_1, X_2)$ have a joint normal distribution with mean $\alpha = (\alpha_1, \alpha_2)$ and variance-covariance matrix Σ -- Then, if we write

$$\Sigma = O\, D\, O^t$$

where we'll assume that Σ is non-singular so that elements of D are positive, we know (or can directly verify) that the inverse of Σ is just $OD^{-1}O^t$

Mahalanobis Distance

Next, we can think of the (squared) Mahalanobis distance from a point m to the center α as taking the usual Euclidean distance between transformed variables

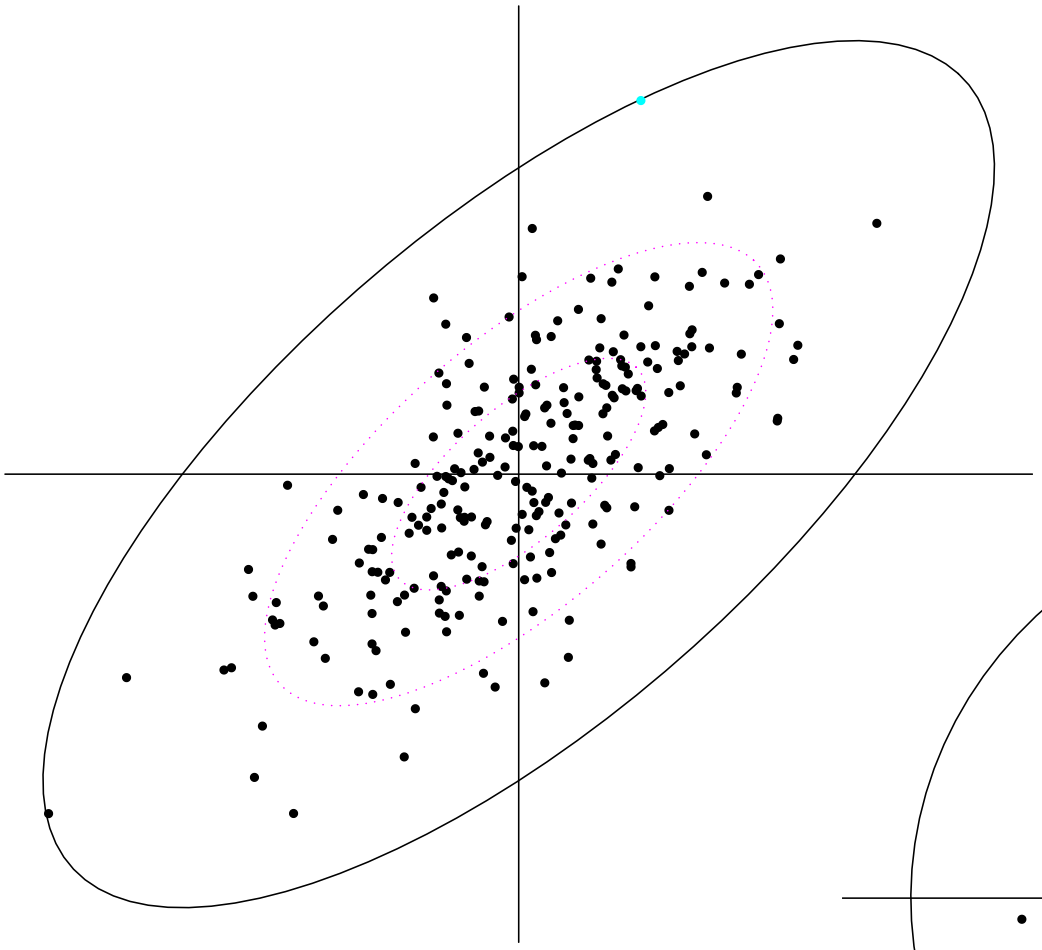
$$\begin{aligned}(\mathsf{m}-\alpha)^{\mathsf{t}}\Sigma^{-1}(\mathsf{m}-\alpha) &= (\mathsf{m}-\alpha)^{\mathsf{t}}\mathsf{O}\mathsf{D}^{-1}\mathsf{O}^{\mathsf{t}}(\mathsf{m}-\alpha)\\ &= (\mathsf{m}-\alpha)^{\mathsf{t}}\mathsf{O}\mathsf{D}^{-1/2}\mathsf{D}^{-1/2}\mathsf{O}^{\mathsf{t}}(\mathsf{m}-\alpha)\\ &= \mathsf{z}^{\mathsf{t}}\mathsf{z}\\ &= \|\mathsf{z}\|^2\end{aligned}$$

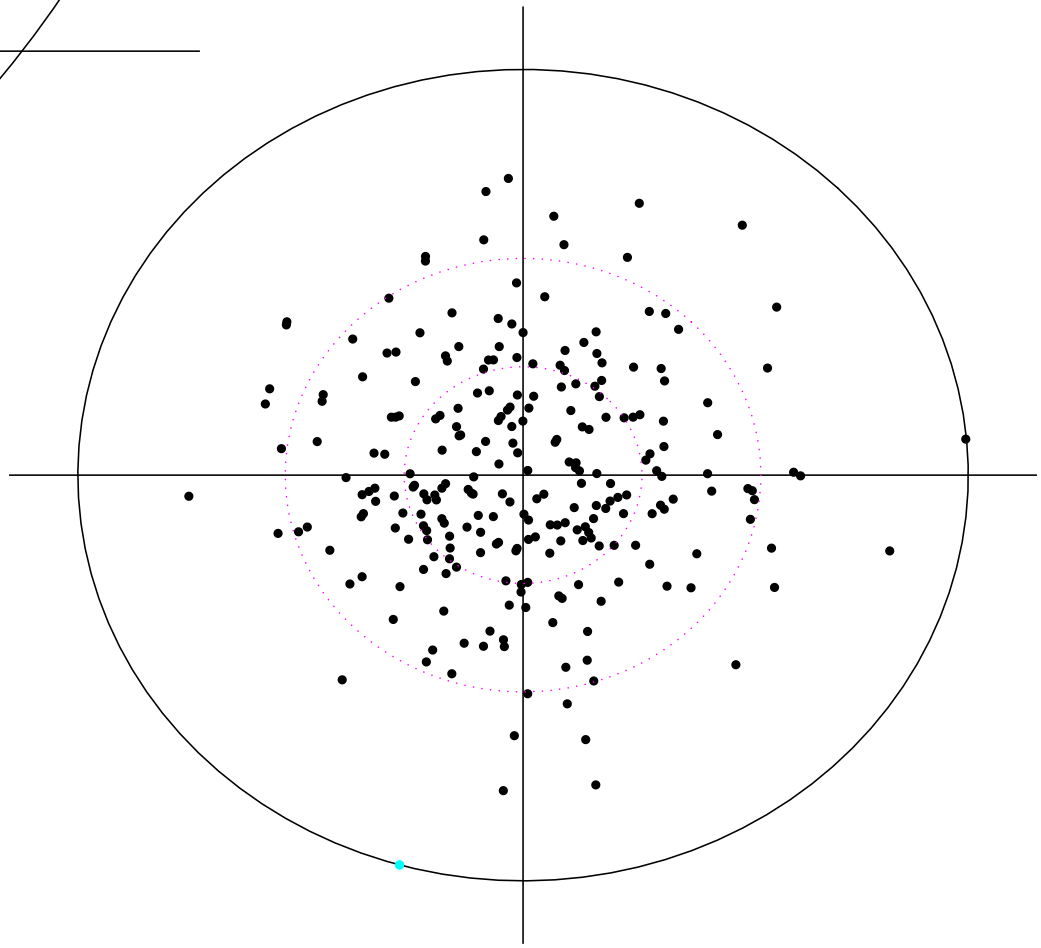
where we have defined $\mathsf{z} = \mathsf{D}^{-1/2}\mathsf{O}^{\mathsf{t}}(\mathsf{m}-\alpha)$

Mahalanobis Distance

Finally, note that if we define $Z = D^{-1/2}O^t(X - \alpha)$, then the vector Z again has a normal distribution but this time with variance-covariance matrix (just multiply it out) the identity

Therefore, the Mahalanobis distance can also be thought of as “whitening” the data before computing a distance, in effect transforming the variables so that they are independent with unit variance





Mahalanobis Distance

Finally, then, returning to our original regression problem, recall that we have assumed we've centered all of our predictor variables -- In that case the columns of our model matrix are of the form

$$\widetilde{M} = [x_1 - \bar{x}_1 | x_2 - \bar{x}_2 | \cdots | x_p - \bar{x}_p]$$

which means that $M^t M/(n-1)$ is really just an estimate of the variance-covariance matrix of our inputs and that the expression

$$(m - \bar{x})^t \left(\frac{\widetilde{M}^t \widetilde{M}}{n-1} \right)^{-1} (m - \bar{x})$$

is the Mahalanobis distance between a new point m and the "center" of our input space $\bar{x} = (\bar{x}_1, \cdots, \bar{x}_p)^t$

Mahalanobis Distance

Now, if we replace $m - \bar{x}$ with, say, the ith row of the model matrix $\widetilde{M}$ (call it $\widetilde{m}_i$), then the expression on the previous page is really just (n-1) times the ith diagonal element of the hat matrix associated with our centered predictors -- That is

$$(n-1)\widetilde{h}_i = (n-1)\widetilde{m}_i^t(\widetilde{M}^t\widetilde{M})^{-1}\widetilde{m}_i = \widetilde{m}_i^t\left(\frac{\widetilde{M}^t\widetilde{M}}{n-1}\right)^{-1}\widetilde{m}_i$$

To relate this to our original predictors, you can show that the ith diagonal element of the hat matrix $\widetilde{h}_i$ associated with the centered variables is just

$$\widetilde{h}_i = h_i - 1/n$$

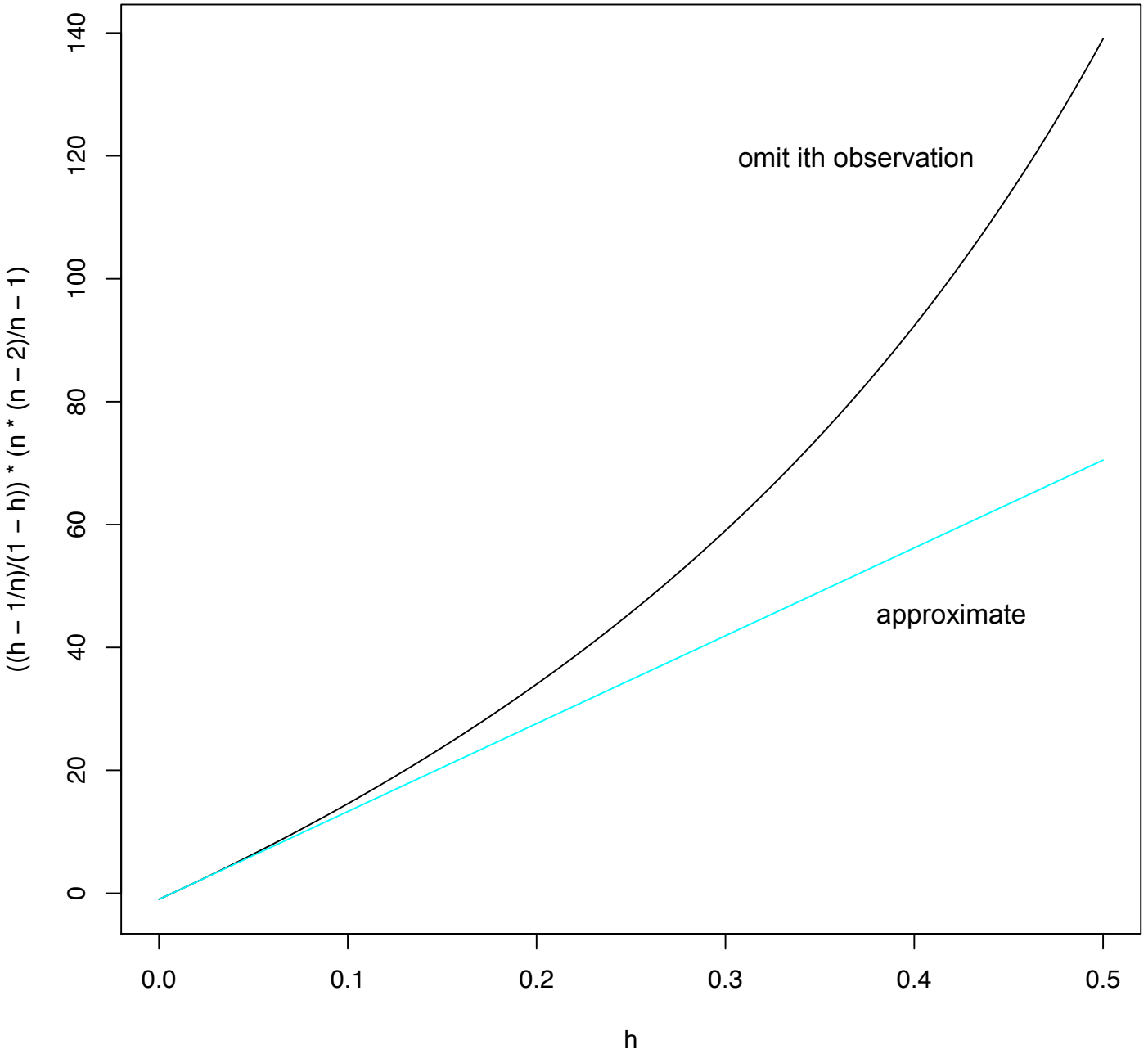
Therefore, $(n-1)(h_i - 1/n)$ can be thought of as the Mahalanobis distance between the ith observation and the “center” of the input space

Mahalanobis Distance

Technically, if we wanted to be a fair bit fussier about this, we would compute the mean of our data and the variance-covariance matrix used in the Mahalanobis distance omitting the ith variable -- Why?

If we do this, a little more algebra suggests that the “true” Mahalanobis distance is still a function of the hat matrix elements, but a slightly more complicated function

$$\left(\frac{h_i - 1/n}{1 - h_i}\right)\frac{n(n-2)}{n-1}$$

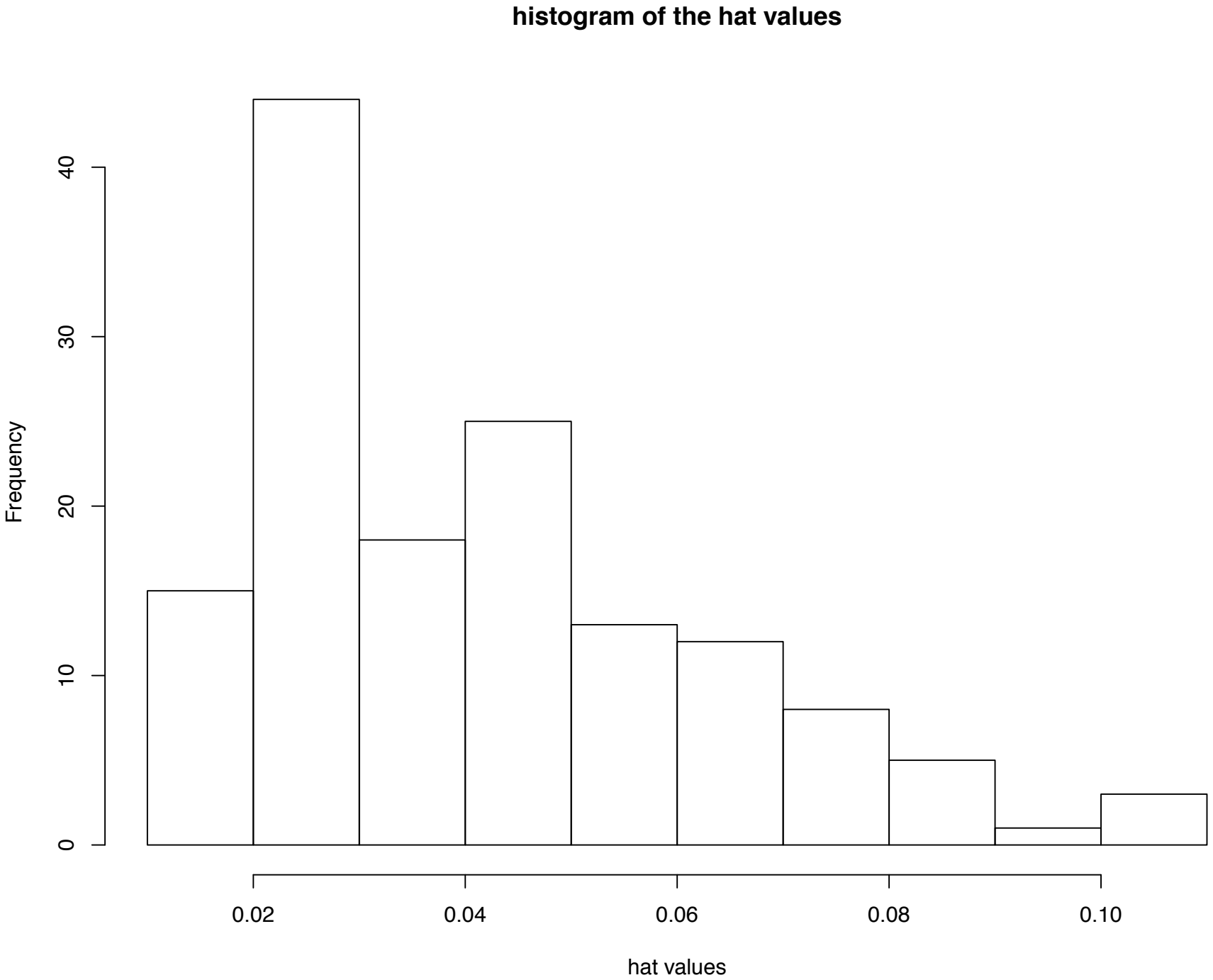

((h – 1/n)/(1 – h)) * (n * (n – 2)/n – 1)
0
20
40
60
80
100
120
140
0.0
0.1
0.2
0.3
0.4
0.5
h
omit ith observation
approximate

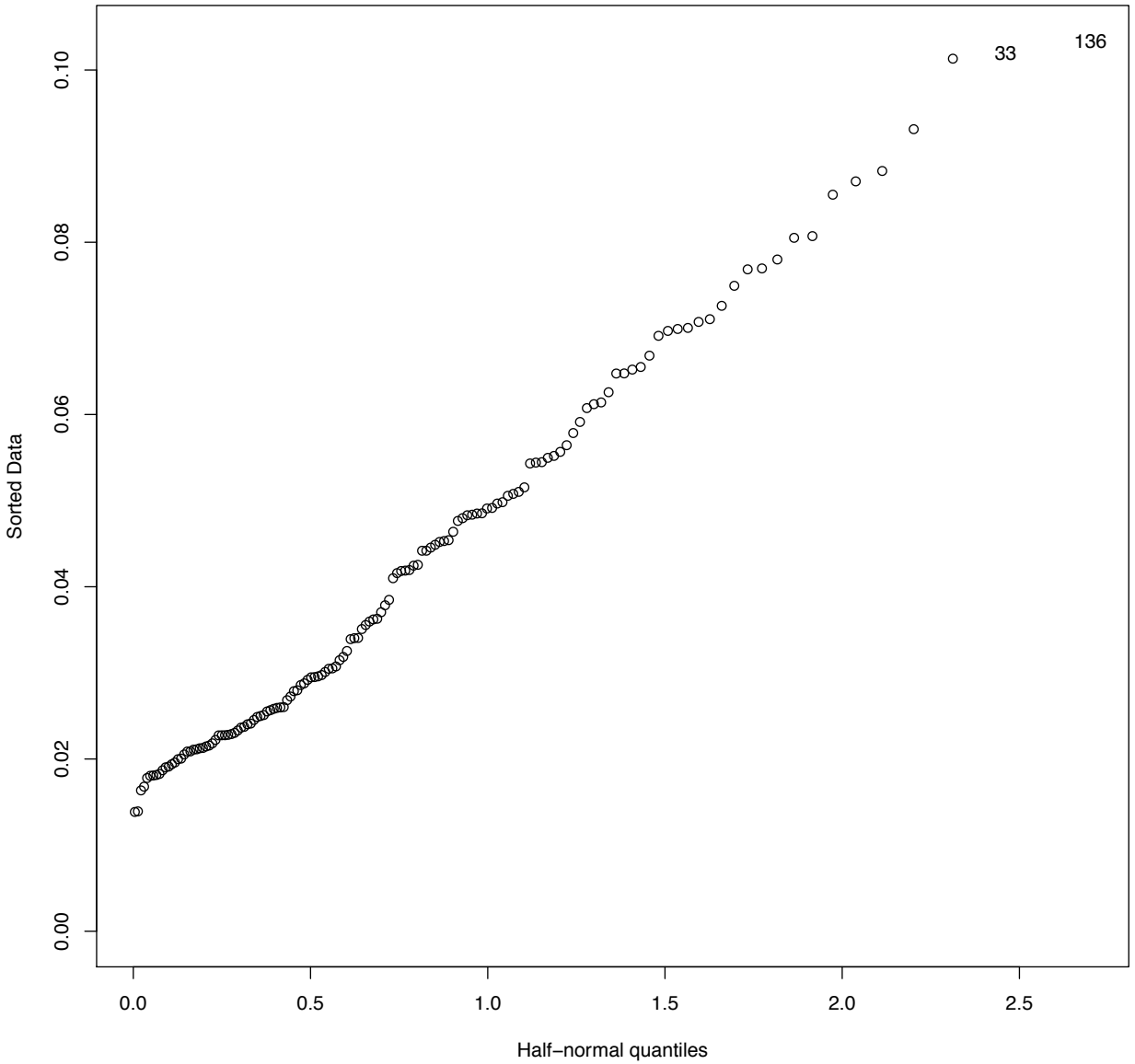
Leverage

Aside from intuitive “cutoff values,” what we really want is to look for points that seem large relative to the others -- For this, a graphical device will do

Faraway in his text on regression suggests using half-normal plots (a construction I am certain you recall from 201a) -- Here we don’t expect the leverage values to be normal and so we’re not looking for a straight line

Instead we want to identify points that are extreme in some way...


histogram of the hat values
Frequency
0
10
20
30
40
0.02
0.04
0.06
0.08
0.10
hat values


136
33
Sorted Data
0.00
0.02
0.04
0.06
0.08
0.10
0.0
0.5
1.0
1.5
2.0
2.5
Half−normal quantiles

Leverage and influence

Using the Mahalanobis interpretation, leverage suggests to us points in the design space that are separate somehow and should be examined more closely -- It is not at all clear that they are affecting the fit in some way

This leads us to the idea of influence -- What happens when we delete a single point from the fit? Ideally, not much should change, but when it does, we refer to these points as being influential

While it seems like a lot of work to assess changes in large data sets (having to leave out points and refit the model each time), it turns out the hat matrix can make these calculations easier...

Fitting the right model

Let’s now fit with the author’s final model -- That is, we include both HDI and the square of HDI for a quadratic fit...

```
# fit without quadratic on hdi for the moment

fit <- lm(ln_death_risk~ln_events+ln_fert+ln_pop+hdi,data=vul)
summary(fit)

# Call:
# lm(formula = ln_death_risk ~ ln_events + ln_fert + ln_pop + hdi,
#    data = vul)
#
# Residuals:
#     Min      1Q  Median      3Q     Max
# -3.4518 -0.7673 -0.1513  0.5669  6.2271
#
# Coefficients:
#             Estimate Std. Error t value Pr(>|t|)
# (Intercept)  -5.3485     1.5175  -3.524 0.000575 ***
# ln_events     1.3708     0.1792   7.649 3.04e-12 ***
# ln_fert       2.1961     0.4614   4.760 4.81e-06 ***
# ln_pop       -0.5672     0.1026  -5.529 1.54e-07 ***
# hdi           1.9922     1.2628   1.578 0.116928
# ---
# Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 #
# Residual standard error: 1.35 on 139 degrees of freedom
# Multiple R-squared: 0.4221,!Adjusted R-squared: 0.4055
# F-statistic: 25.38 on 4 and 139 DF,  p-value: 8.522e-16
```

```
fit <- lm(ln_death_risk~ln_events+ln_fert+ln_pop+hdi+I(hdi^2),data=vul)
summary(fit)

# Call:
# lm(formula = ln_death_risk ~ ln_events + ln_fert + ln_pop + hdi +
#     I(hdi^2), data = vul)
#
# Residuals:
#      Min       1Q   Median       3Q      Max
# -3.81655 -0.80298 -0.04575  0.63866  5.60679
#
# Coefficients:
#              Estimate Std. Error t value Pr(>|t|)
# (Intercept) -10.92243    1.81119  -6.031 1.42e-08 ***
# ln_events     1.42774    0.16648   8.576 1.79e-14 ***
# ln_fert       1.47558    0.45232   3.262  0.00139 **
# ln_pop       -0.56450    0.09507  -5.938 2.22e-08 ***
# hdi          25.06179    4.86656   5.150 8.81e-07 ***
# I(hdi^2)    -18.89905    3.86980  -4.884 2.84e-06 ***
# ---
# Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1 #
# Residual standard error: 1.251 on 138 degrees of freedom
# Multiple R-squared: 0.5073,!Adjusted R-squared: 0.4894
# F-statistic: 28.41 on 5 and 138 DF,  p-value: < 2.2e-16
```

Orthogonality

Last time we defined orthogonal vectors as those having zero inner product -- That is, we say that x_1 and x_2 are orthogonal if $x_1^t x_2 = 0$

We probably remember a basic result from geometry that if θ is the angle between x_1 and x_2 then

$$\cos\theta = \frac{\sum_i x_{i1}x_{i2}}{\sqrt{\sum_i x_{i1}^2}\sqrt{\sum_i x_{i2}^2}}$$

so that x_1 and x_2 are orthogonal if $\sum_i x_{i1}x_{i2} = 0$

Orthogonality

If we assume further that the length of each vector, $\|x_1\|$ and $\|x_2\|$ are both 1, then we'll say that the vectors are orthonormal

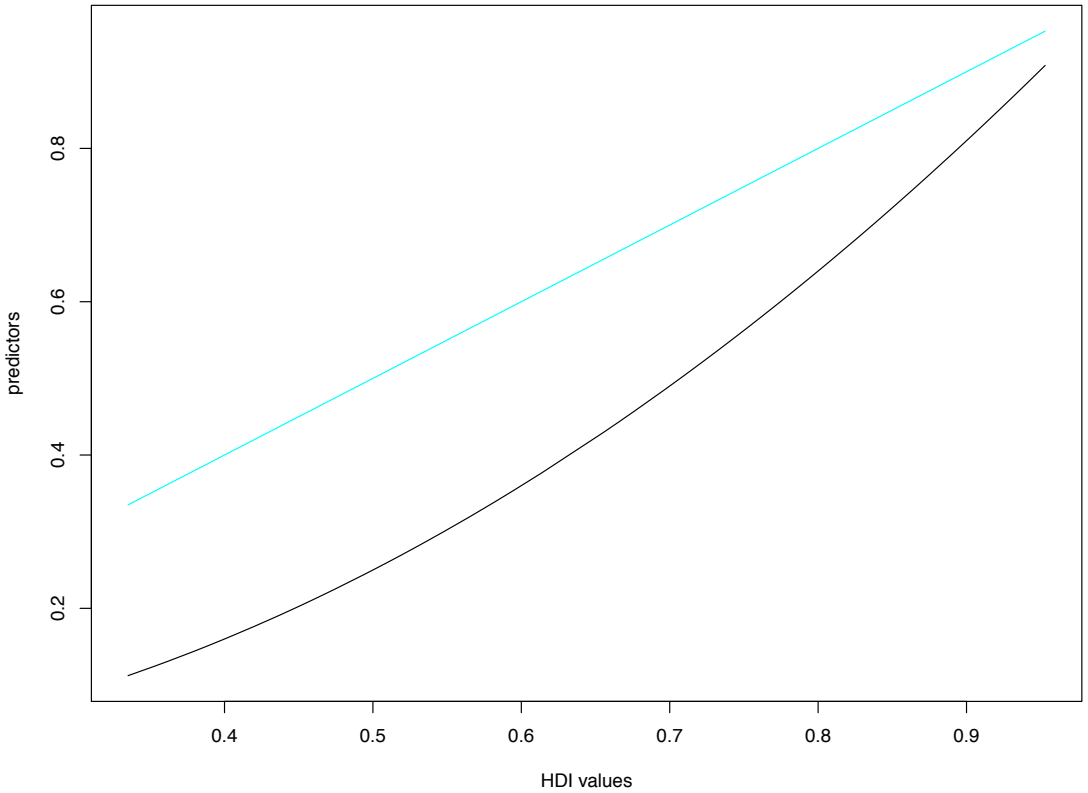
If a miracle (of design) happens and we have orthonormal predictors in a linear model, what does that say about our estimates? Do things simplify?

Orthogonality

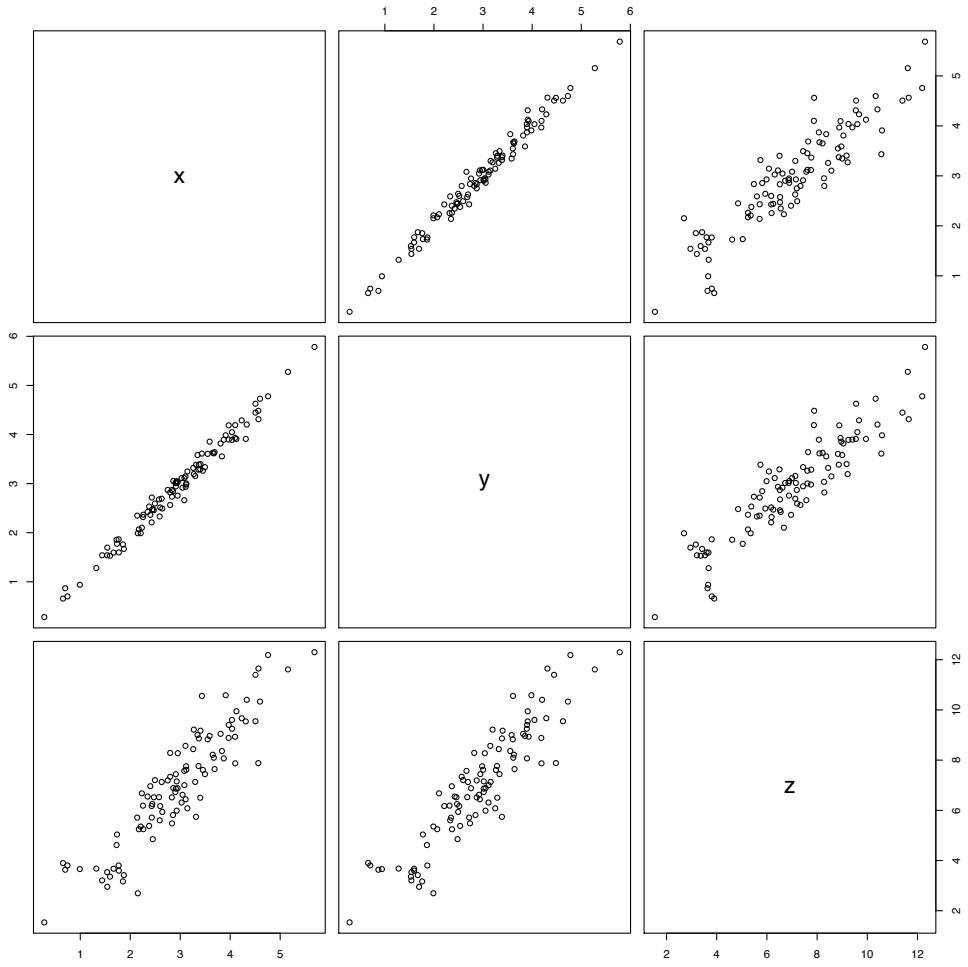
What strange thing happened with the HDI predictor when we included its square? Why?

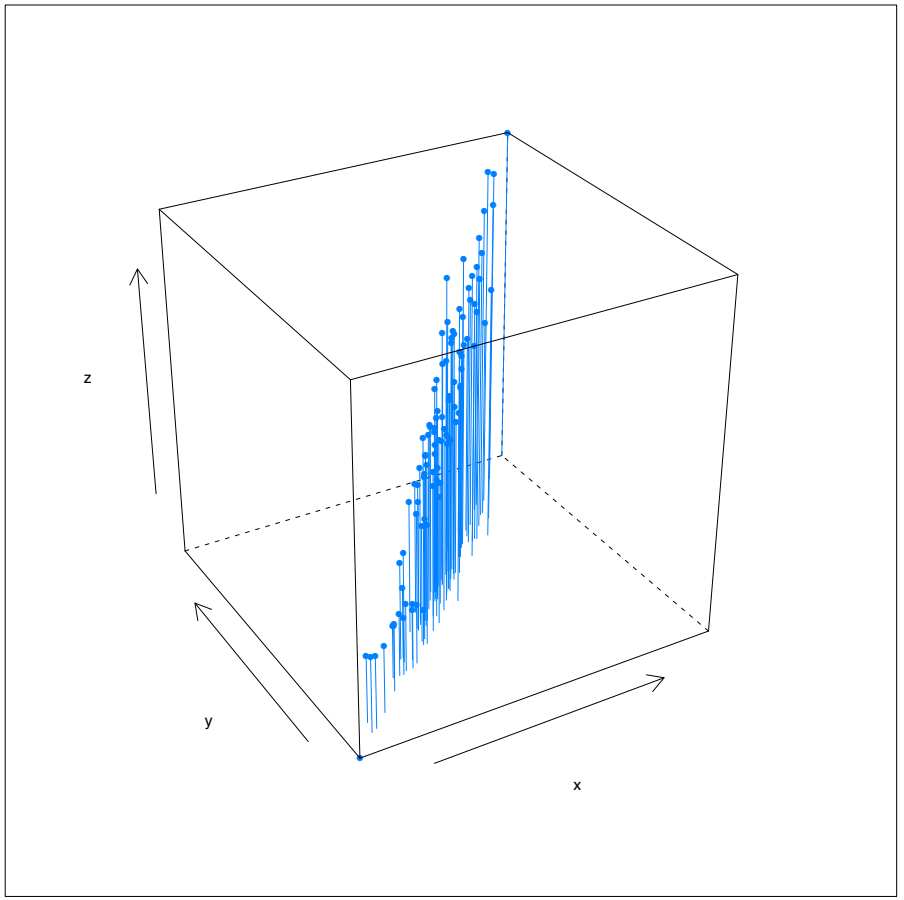
Orthogonality

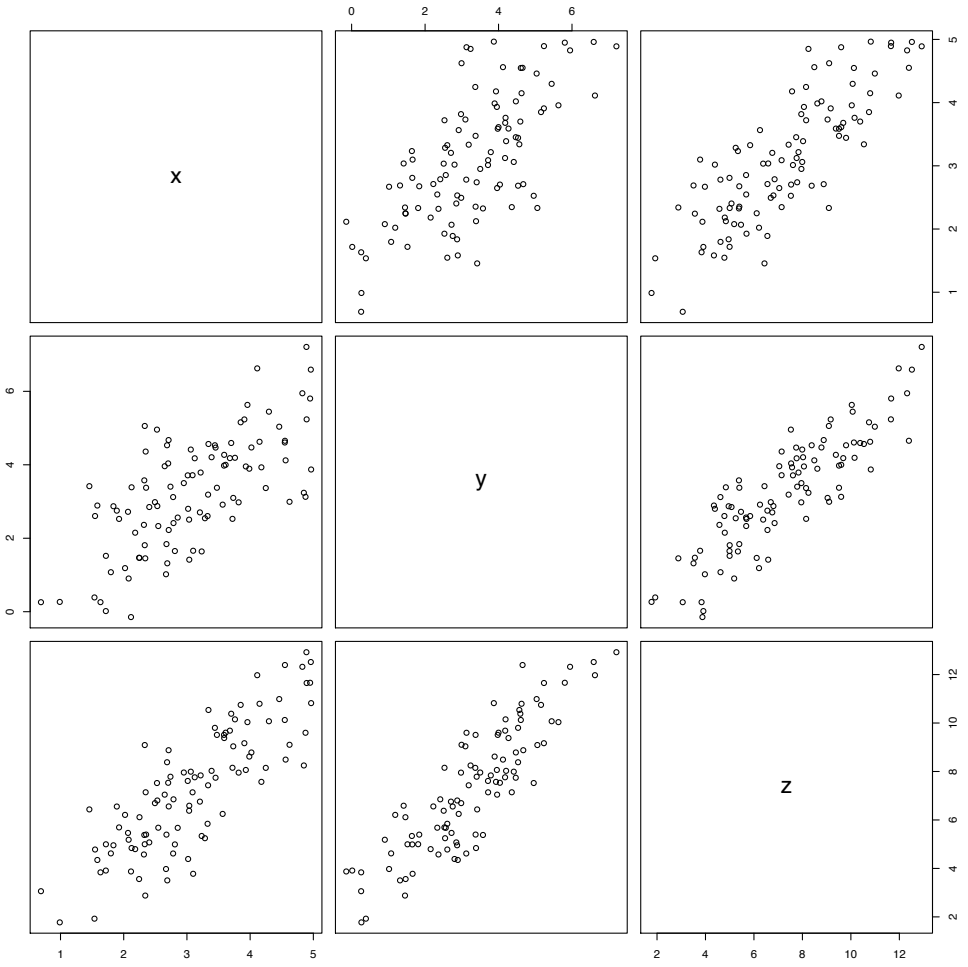
It turns out that HDI and its square are very highly correlated (0.992 to be more or less precise) -- Might this cause a problem?

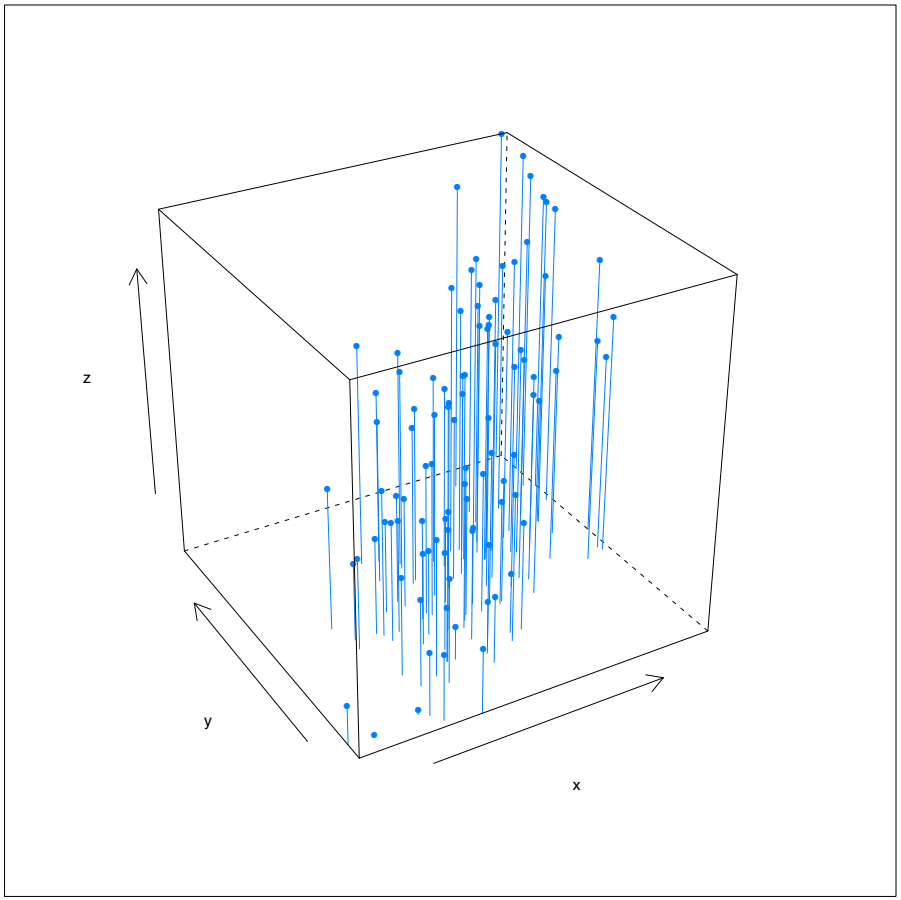


```
library(lattice)    # load library with 3d plotting tools

# make two data sets, the first having highly
# correlated predictors
t1 = rnorm(100,m=3)
t2 = t1+rnorm(100,sd=0.15)
t3 = 1+t2+t1 + rnorm(100)
simu1 = data.frame(x=t1,y=t2,z=t3)
pairs(simu1)
cloud(z~x*y,data=simu1,screen=list(z=30,x=-60),type=c("h","p"),pch=16)

# and the second having less correlated predictors
t1 = rnorm(100,m=3)
t2 = t1+rnorm(100,sd=1)
t3 = 1+t2+t1 + rnorm(100)
simu2 = data.frame(x=t1,y=t2,z=t3)
pairs(simu2)
cloud(z~x*y,data=simu2,screen=list(z=30,x=-60),type=c("h","p"),pch=16)

# now, let's fit a linear model in each case
# and see what's different
fit1 = lm(z~x+y,data=simu1)
summary(fit1)
fit2 = lm(z~x+y,data=simu2)
summary(fit2)
# what do you notice?
```


x
y
z


z
y
x


x
y
z


z
y
x

```
> fit1 = lm(z~x+y,data=simu1)
> summary(fit1)

Call:
lm(formula = z ~ x + y, data = simu1)

Residuals:
     Min       1Q   Median       3Q      Max
-2.49628 -0.56111 -0.02548  0.67046  2.33352

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.9201     0.2861   3.216  0.00177 **
x             0.9499     0.6736   1.410  0.16166
y             1.1196     0.6736   1.662  0.09972 .
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9342 on 97 degrees of freedom
Multiple R-squared: 0.8397,   Adjusted R-squared: 0.8364
F-statistic:   254 on 2 and 97 DF,  p-value: < 2.2e-16
```

```
> fit2 = lm(z~x+y,data=simu2)
> summary(fit2)

Call:
lm(formula = z ~ x + y, data = simu2)

Residuals:
     Min       1Q   Median       3Q      Max
-2.37281 -0.67861 -0.06506  0.83082  2.15703

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.72558    0.32167   2.256   0.0263 *
x            1.11074    0.13664   8.129 1.43e-12 ***
y            0.95760    0.08906  10.752  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.9827 on 97 degrees of freedom
Multiple R-squared: 0.8556,   Adjusted R-squared: 0.8526
F-statistic: 287.4 on 2 and 97 DF,  p-value: < 2.2e-16
```

Variance inflation factors

The stability issue, then, comes down to how correlated a predictor is with others in the model -- Put another way, how well can we each predictor be explained by the others?

Operationally, we could consider regressing each predictor on the others and examining some goodness of fit measure like, say, the squared correlation coefficient -- This is the rationale behind the so-called variance inflation factor

$$\text{VIF} = \frac{1}{1 - R^2}$$

What is the smallest value the VIF can be? The largest?

```
# standardizing variables
# let's work with the vulnerability data

load(url("http://www.stat.ucla.edu/~cocteau/stat201b/vulnerability.RData"))
names(vul)

# [1] "country_name"  "ln_urb"        "ln_events"     "ln_fert"
# [5] "hdi"           "ln_pop"        "ln_death_risk" "death_risk"

fit = lm(hdi~ln_events+ln_fert+ln_pop,data=vul)

# and extract the r2 value
r2 = summary(fit)$r.squared
varinf = 1/(1-r2)

# now let's try a function that does it
# for us; julian faraway used to be a
# prof at michigan and this is his library
# of regression tools

library(faraway)
vif(vul[,3:6])

# should match what we had for hdi!
```

Variance inflation factors

Given their name, you won't be surprised that we can relate the standard error of a regression estimate to the associated VIF

$$\text{se}(\widehat{\beta}_j) = \widehat{\sigma}\sqrt{\frac{\text{VIF}_j}{\sum_i (x_{ij} - \bar{x}_j)^2}}$$

Therefore, the larger the VIF for some predictor (the better "explained" it is by the other variables we have in our model), the greater the standard error of the regression estimate

Variance inflation factors

Finally, we return to our simulated examples; in the first case, the predictors are highly correlated, while in the second they are only moderately so

This, then, explains the crazy behavior we were seeing; the instability is dramatically inflating our standard errors

People sometimes provide “rules” about when you’re facing a serious problem (say VIF > 10), bit I prefer to view this as another diagnostic

```
> vif(simu1[,-3])
       x        y
54.07696 54.07696

> vif(simu2[,-3])
       x        y
1.902763 1.902763

> vif(cbind(vul[,c("ln_events","ln_fert","ln_pop","hdi")],vul$hdi^2))

ln_events   ln_fert    ln_pop       hdi vul$hdi^2
 2.433686  4.076010  2.460708 65.156728 71.792326
```

Orthogonal polynomials

The typical approach to fitting with polynomials is not to toss in higher-order monomials, but instead to orthogonalize the variables as they are introduced into the regression

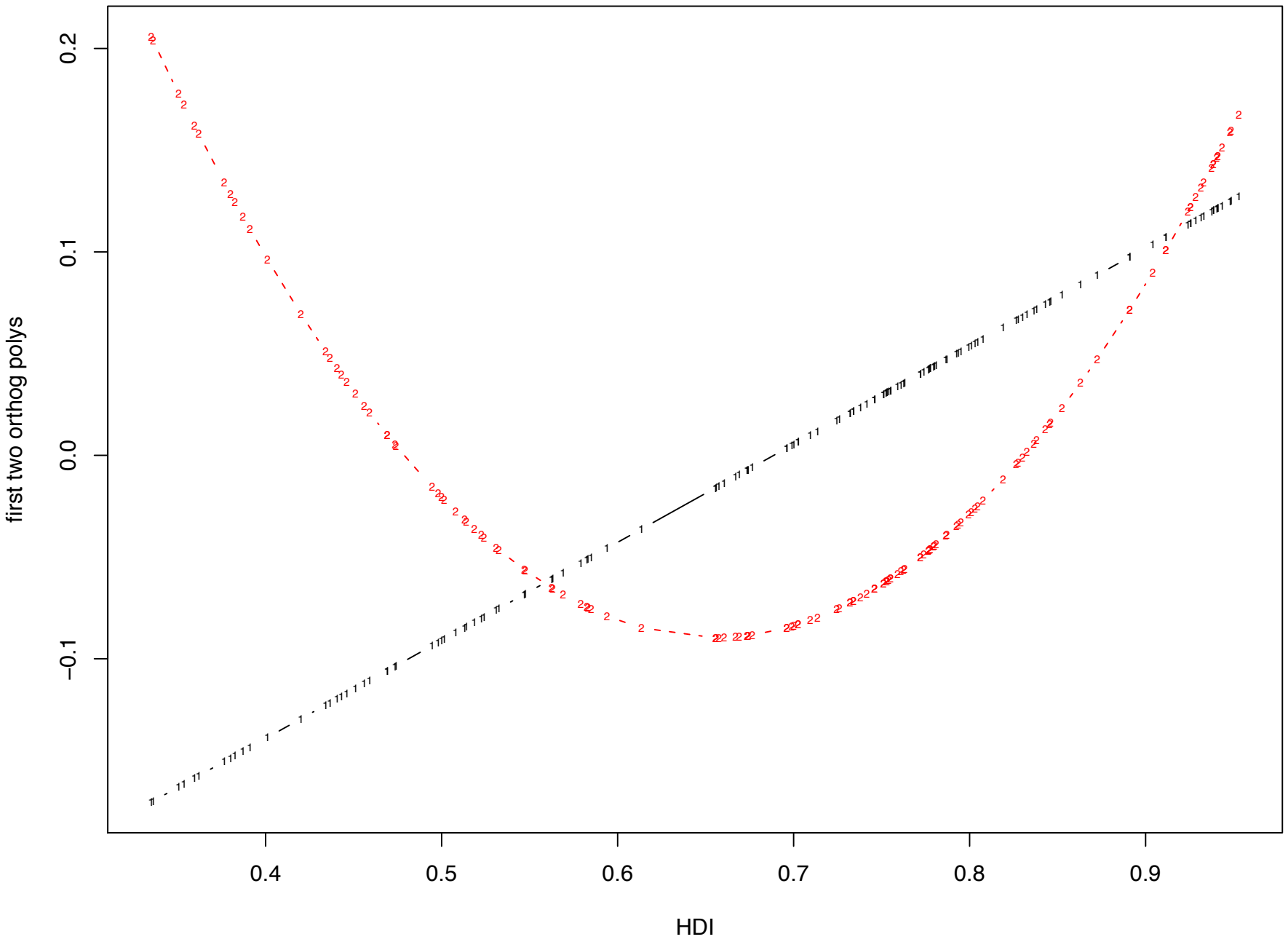
We know that the space of quadratic polynomials in HDI are spanned by three columns, say, $1, \mathrm{HDI}, \mathrm{HDI}^2$

Suppose we add them into the model sequentially, each time adding not the whole variable, but instead the residuals after having regressed out the terms that came before

Orthogonal polynomials

So the first term to enter is simply the constant vector -- To regress the variable HDI onto this “variable” means...

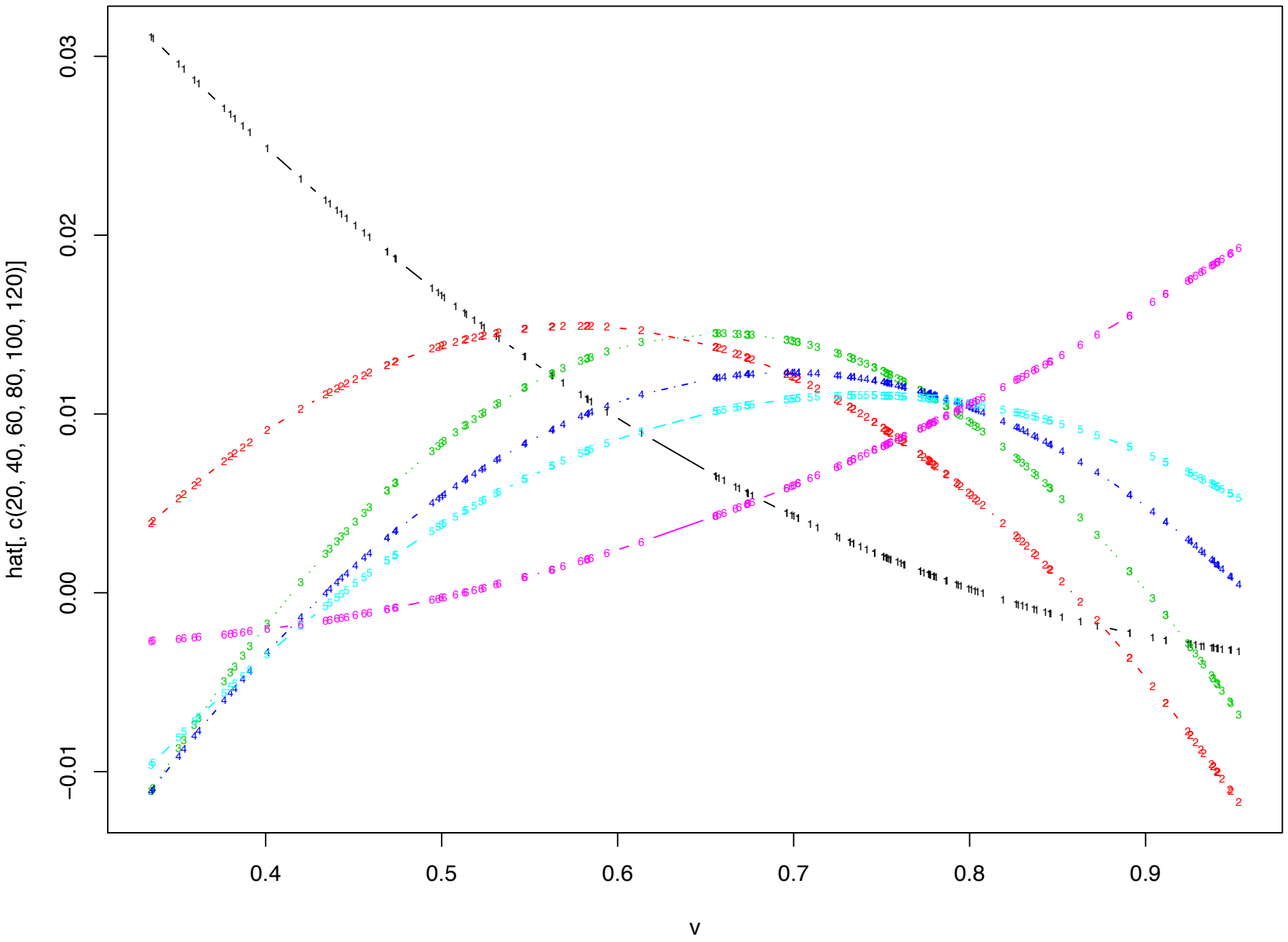
```
h <- sort(vul$hdi)

h0 <- rep(1,length(h))
h0 <- h0/sqrt(sum(h0^2))

h1 <- h
h1 <- h1-mean(h1)
h1 <- h1/sqrt(sum(h1^2))

h2 <- h^2
h2 <- residuals(lm(h2~h0+h1))
h2 <- h2/sqrt(sum(h2^2))

matplot(h,cbind(h1,h2),type="b",cex=0.5)

t(cbind(h0,h1,h2))%*%cbind(h0,h1,h2)

#               h0            h1            h2
# h0  1.000000e+00 -3.625572e-16 -3.556183e-16
# h1 -3.625572e-16  1.000000e+00  2.567391e-16
# h2 -3.556183e-16  2.567391e-16  1.000000e+00

# compare to poly(h,2)

vif(cbind(vul[,c("ln_events","ln_fert","ln_pop")],poly(vul$hdi,2)))

# ln_events   ln_fert    ln_pop         1         2
#  2.433686  4.076010  2.460708  4.092598  1.145813
```

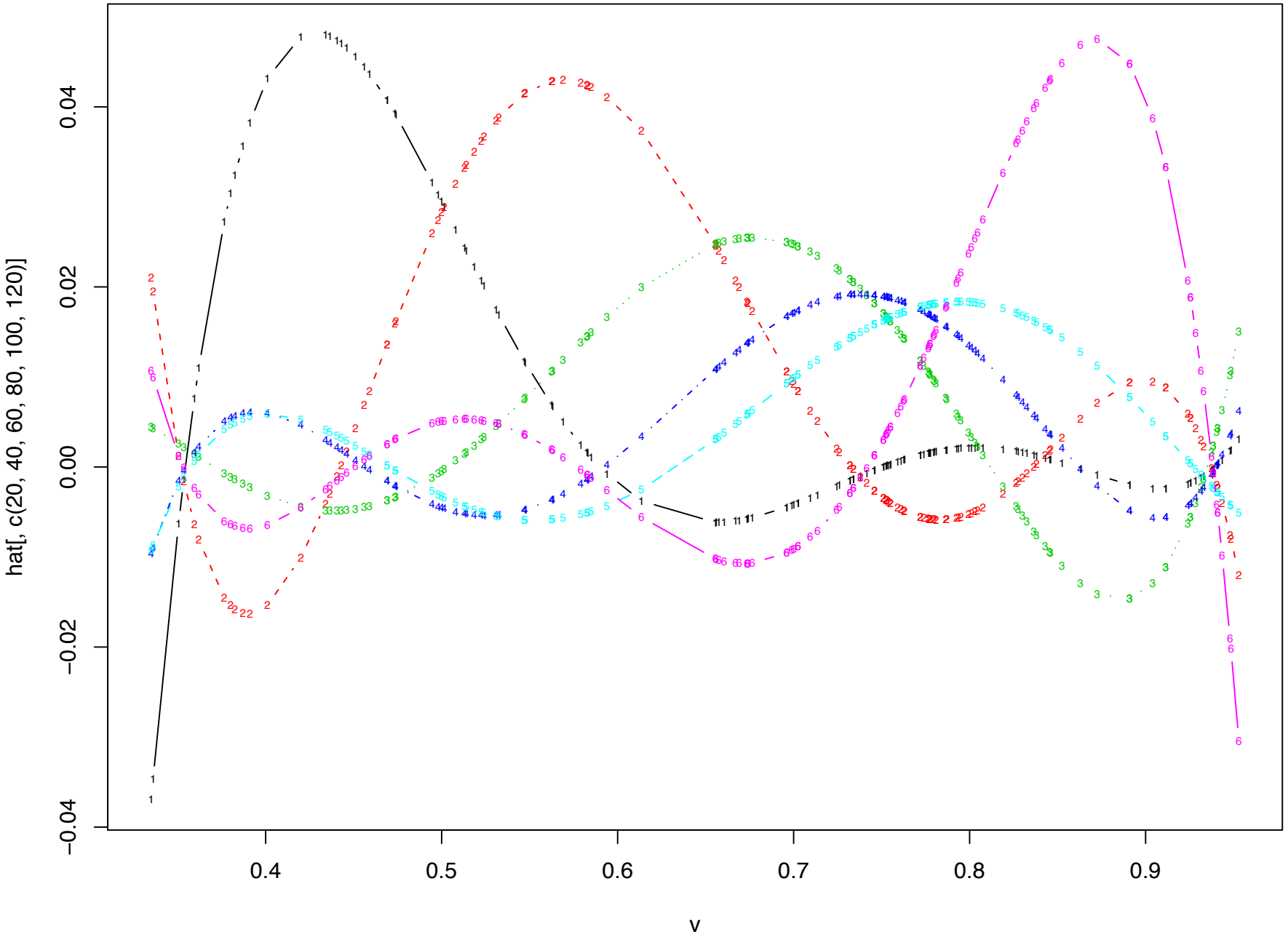

first two orthog polys
0.2
0.1
0.0
−0.1
0.4
0.5
0.6
0.7
0.8
0.9
HDI

Aside: Equivalent kernels

We can use what we know about the hat matrix to help us understand what polynomial fitting is doing -- On the next two slides we present the rows of the hat matrix (20, 40, 60, 80, 100, 120) corresponding to a quadratic fit of HDI and a quintic fit

What do you notice?


hat[, c(20, 40, 60, 80, 100, 120)]
0.03
0.02
0.01
0.00
−0.01
0.4
0.5
0.6
0.7
0.8
0.9
v


hat[, c(20, 40, 60, 80, 100, 120)]
0.04
0.02
0.00
−0.02
−0.04
0.4
0.5
0.6
0.7
0.8
0.9
v

Equivalent kernels

While this is skipping ahead a little, the rows of the hat matrix formed when performing a regression with polynomials, or piecewise polynomials or splines can be thought of as kernel functions

When we use linear regression and polynomials the projection or hat matrix that takes $\widehat{\mu} = \mathrm{M}(\mathrm{M}^{\mathrm{t}}\mathrm{M})^{-1}\mathrm{M}\mathrm{y} = \mathrm{H}\mathrm{y}$ -- It turns out that a large class of so-called linear smoothers are of the general form $\widehat{\mu} = \mathrm{S}\mathrm{y}$

The ith row of S then represent weights applied to the observations in y to form the estimated $\widehat{\mu}_{\mathrm{i}}$

Orthogonal polynomials

Finally, let's have a look at the fit using these variables instead of HDI and its square -- What do you notice?

```
> fit <- lm(ln_death_risk~ln_events+ln_fert+ln_pop+poly(hdi,2),data=vul)
> summary(fit)

Call:
lm(formula = ln_death_risk ~ ln_events + ln_fert + ln_pop + poly(hdi,
    2), data = vul)

Residuals:
     Min       1Q   Median       3Q      Max
-3.81655 -0.80298 -0.04575  0.63866  5.60679

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   -3.19131    0.72498  -4.402 2.13e-05 ***
ln_events      1.42774    0.16648   8.576 1.79e-14 ***
ln_fert        1.47558    0.45232   3.262  0.00139 **
ln_pop        -0.56450    0.09507  -5.938 2.22e-08 ***
poly(hdi, 2)1  0.65104    2.53050   0.257  0.79735
poly(hdi, 2)2 -6.53905    1.33895  -4.884 2.84e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.251 on 138 degrees of freedom
Multiple R-squared: 0.5073,   Adjusted R-squared: 0.4894
F-statistic: 28.41 on 5 and 138 DF,  p-value: < 2.2e-16
```

Regression by successive orthogonalization

The stepwise process we followed to successively orthogonalize $1, \text{HDI}, \text{HDI}^2$ can be applied generally to any set of predictors -- The idea would be to sidestep the stability issues with correlated input variables by orthogonalizing them before they are added to the model

This process is known in linear algebra texts as Gram-Schmidt orthogonalization and is used to build an orthogonal basis for a given subspace (say the column space of a matrix)

Orthogonal predictors

Suppose we were given orthogonal predictors $q_1, \ldots, q_p$, we can solve the least squares problem easily -- If we let $Q = [q_1, \ldots, q_p]$ be our new model matrix (Q because it sorta looks like an O which would stand for “orthogonal” -- Seriously) and want to find a vector α to minimize

$$\|y - Q\alpha\|^2$$

Then our solution is (symbolically)

$$\widehat{\alpha} = (Q^t Q)^{-1} Q^t y$$

But because $q_1, \ldots, q_p$ are orthogonal, $(Q^t Q)$ is diagonal with entries $q_j^t q_j = \|q_j\|^2$ which means the components of our solution vector are simply

$$\widehat{\alpha}_j = q_j^t y / q_j^t q_j$$

Gram-Schmidt

Therefore, in a regression context, we can think of the Gram-Schmidt process in the following way

1. Set $z_1 = (1, \ldots, 1)$

2. For j=2,...,p

   Regress m_j on $z_1, \ldots, z_{j-1}$ to produce the coefficients

$$\widehat{\gamma}_{lj} = \frac{z_l^t m_j}{z_l^t z_l} = \frac{\sum_i z_{il} m_{il}}{\sum_i z_{il}^2} \quad \text{for} \quad l = 1, \ldots, j-1$$

   and form the new basis vector

$$z_j = m_j - (\widehat{\gamma}_{1j} z_1 + \cdots + \widehat{\gamma}_{j-1,j} z_{j-1})$$

Unrolling

Rearranging the residuals in step (2) we find that

$$
\begin{aligned}
m_1 &= z_1 \\
m_2 &= z_2 - \widehat{\gamma}_{12} z_1 \\
m_3 &= z_3 - \widehat{\gamma}_{23} z_2 - \widehat{\gamma}_{13} z_1 \\
&\vdots \\
m_p &= z_p - \widehat{\gamma}_{p-1,p} z_{p-1} - \cdots - \widehat{\gamma}_{1p} z_1
\end{aligned}
$$

And so each m_j is a linear combination of the z_l for $l \leq j$; since all the z_l are orthogonal, they form a basis for the column space of M

Aside: Invertible transformations

Again, this process allows us to construct a new basis for the column space of M, one that is orthogonal -- We can then easily solve for the least squares projection $\widehat{\mu}$ of y

Note, however, that we are no longer modeling with the same set of basis functions (predictor variables) we started with, and any interpretation based on the fitted coefficients in our model will change

In general, if we use an invertible p-by-p matrix A to create a new set of predictor variables from our old ones $\mathrm{M}' = \mathrm{MA}$, then the value of α (a vector) that minimizes

$$\|\mathrm{M}'\alpha - \mathrm{y}\|^2 = \|\mathrm{MA}\alpha - \mathrm{y}\|^2$$

is simply $\widehat{\alpha} = \mathrm{A}^{-1}\widehat{\beta}$, a transformation of our previous OLS estimates

Because the column spaces of M and MA are the same, the projection of y into this space, $\widehat{\mu}$, is unchanged by the transformation -- In some sense, then, you might argue that this property makes the conditional means μ more sensible parameters to think about than the $\beta'\mathrm{s}$

An observation

Let $\widehat{\beta}$ be the least squares estimate of β, so that it minimizes $\|y - M\beta\|^2$, and write out our expression for the estimated conditional means

$$\widehat{\mu} = \widehat{\beta}_1 m_1 + \cdots + \widehat{\beta}_p m_p$$

We can then used the unrolled expressions on the previous slide to rewrite $\widehat{\mu}$ in terms of the orthogonal basis functions $q_1, \ldots, q_p$

$$\widehat{\mu} = \widehat{\beta}_1 z_1 + \widehat{\beta}_2 (z_2 - \widehat{\gamma}_{12} z_1) + \cdots + \widehat{\beta}_p (z_p - \widehat{\gamma}_{p-1,p} z_{p-1} - \cdots - \widehat{\gamma}_{1p} z_1)$$

But notice that z_p only appears in $m_p$and with a coefficient 1 -- That means that $\widehat{\beta}_p$ is the same as the regression coefficient for z_p when we regress y on the orthogonal basis $z_1, \ldots, z_p$

Therefore, we've shown that

$$\widehat{\beta}_p = z_p^t y / z_p^t z_p = z_p^t y / \|z_p\|^2$$

An observation

This means that the regression coefficient $\widehat{\beta}_p$ is the result of a univariate regression of the response y on z_p where z_p is the residual after regressing m_p on the previous p-1 predictors $m_1, \ldots, m_{p-1}$

Of course there's nothing special about the ordering of our predictor variables and we could have arranged any of the p predictors to be last...

In general

The jth regression coefficient $\widehat{\beta}_j$ is the univariate regression of y on a vector z where z is the residual after regressing m_j on $m_1, \ldots, m_{j-1}, m_{j+1}, \ldots, m_p$

It's simple to show that since $\widehat{\beta}_j = z^t y / z^t z = z^t y / \|z\|^2$

$$\text{var}\ \widehat{\beta}_j = \frac{\sigma^2}{\|z\|^2}$$

This means that if a predictor variable is "well explained" by other variables in the model, we will not be able to estimate it stably

Aside: VIF

This also gives us a proof of an expression involving the VIF we saw earlier --
Let z be the residual after regressing m_j on the remaining $m_1, \ldots, m_{j-1}, m_{j+1}, \ldots, m_p$
and let R^2 denote the associated (unadjusted) correlation coefficient

Then

$$1 - R^2 = 1 - \left(1 - \frac{\sum_i (m_{ij} - \overline{m}_j)^2}{\|z\|^2}\right) = \frac{\sum_i (m_{ij} - \overline{m}_j)^2}{\|z\|^2}$$

So that the VIF for the jth variable is just

$$VIF_j = \frac{\|z\|^2}{\sum_i (m_{ij} - \overline{m}_j)^2}$$

Therefore, we can write the variance on the previous slide as

$$\frac{\sigma^2}{\|z\|^2} = \frac{\sigma^2}{\|z\|^2} \frac{\sum_i (m_{ij} - \overline{m}_j)^2}{\sum_i (m_{ij} - \overline{m}_j)^2} = \frac{\sigma^2 \ VIF_j}{\sum_i (m_{ij} - \overline{m}_j)^2}$$

A second observation

While we have presented the Gram-Schmidt algorithm as a process, we have already seen hints of other representations -- In particular, using the unrolled expression from several slides back we can write

$$M = Z\Gamma$$

where Z has the columns $z_1, \ldots, z_p$ and Γ is an upper-triangular matrix with entries $\widehat{\gamma}_{ij}$ -- Introducing a diagonal matrix D where the jth element is $\|z_j\|^2 = z_j^t z_j$ we can rewrite this expression as

$$M = (ZD^{-1})D\Gamma = QR$$

The QR decomposition

The QR decomposition breaks a matrix M into an n-by-p orthogonal matrix Q and a p-by-p upper triangular matrix R -- This decomposition is unique and it represents one convenient orthogonal basis for the column space of M

With this representation, we can show that the least squares estimates are given by the somewhat simpler expressions

$$\widehat{\beta} = \mathrm{R}^{-1}\mathrm{Q}^{\mathrm{t}}\mathrm{y} \quad \text{and} \quad \widehat{\mu} = \mathrm{Q}\mathrm{Q}^{\mathrm{t}}\mathrm{y}$$

Unlike inverting $\mathrm{M}^{\mathrm{t}}\mathrm{M}$ this form of the solution is easy because R is an upper-triangular matrix -- This is how R (now the programming environment) solves its least squares problems

Modified Gram-Schmidt

We told you that solving the normal equations, $\mathrm{M}^{\mathrm{t}}\mathrm{M}\beta = \mathrm{M}^{\mathrm{t}}\mathrm{y}$, is generally viewed as an unwise approach because of numerical difficulties -- Roundoff errors can, for example, accumulate so that your computation could be somewhat off from the “true” or exact arithmetic solution

Using Gram-Schmidt to create the QR decomposition is also viewed as being somewhat unstable -- Instead people often appeal to the so-called modified Gram-Schmidt procedure

Modified Gram-Schmidt

On the previous slides, we constructed an orthogonal basis stepwise, at step j forming an orthogonal basis $z_1, \ldots, z_{j-1}$ and projecting m_j into the orthogonal complement of their span

That is, we projected into the column space of $z_1, \ldots, z_{j-1}$ and looked at the residual from this fit

Alternately, we could project m_j first into the orthogonal complement of the space spanned by z_1 (look at the residual $m_j - (z_1^t m_j / z_1^t z_1) z_1$) and then project this into the orthogonal complement of the space spanned by z_2 and so on

This alteration is at the core of the modified Gram-Schmidt procedure...

Classical and Modified Gram-Schmidt

The change is best illustrated below

Set $z_1 = (1, \ldots, 1)$

For j=2,...,p

Set $z_j = m_j$

For l = 1,...,j-1

$$\begin{cases} \alpha = z_l^t m_j / z_l^t z_l \ (\text{CGS}) \\ \alpha = z_l^t z_j / z_l^t z_l \ (\text{MGS}) \end{cases}$$

$$z_j = z_j - \alpha z_l$$

A diagnostic

This view of the jth multiple regression coefficient $\widehat{\beta}_j$ as a simple regression onto a new predictor variable, a residual removing the effect of the other predictors in the model, provides us with another diagnostic plot that is commonly discussed in the literature

The so-called added variable plot or partial regression plot is designed to examine the contribution of m_j to the regression, given that the remaining predictors have already been included -- It is said to be an improvement over the residual plots we made last time because it isolates the effects of a single variable

As a graphic, the added variable plot relates two sets of residuals -- Let

$$M_{[j]} = [m_1, \ldots, m_{j-1}, m_{j+1}, \ldots, m_p]$$

be the model matrix removing the jth variable and let (symbolically)

$$\widehat{\epsilon}_{[j]} = y - M_{[j]}(M_{[j]}^t M_{[j]})^{-1} M_{[j]}^t y \quad \text{and} \quad \widehat{z}_j = m_j - M_{[j]}(M_{[j]}^t M_{[j]})^{-1} M_{[j]}^t m_j$$

Added variable plots

In a very real sense, the plot exhibits the regression problem “felt” by the coefficient $\widehat{\beta}_j$ -- What properties do you expect this plot to have?

Added variable plots

First, the slope of the regression line of $\widehat{\epsilon}_{[j]}$ on $\widehat{z}_j$ is exactly $\widehat{\beta}_j$ -- We derived this in the last few slides

Next, the residuals around the line are exactly the residuals from the full fit -- And in this sense we "see" the actual regression that produces the coefficient estimate $\widehat{\beta}_j$

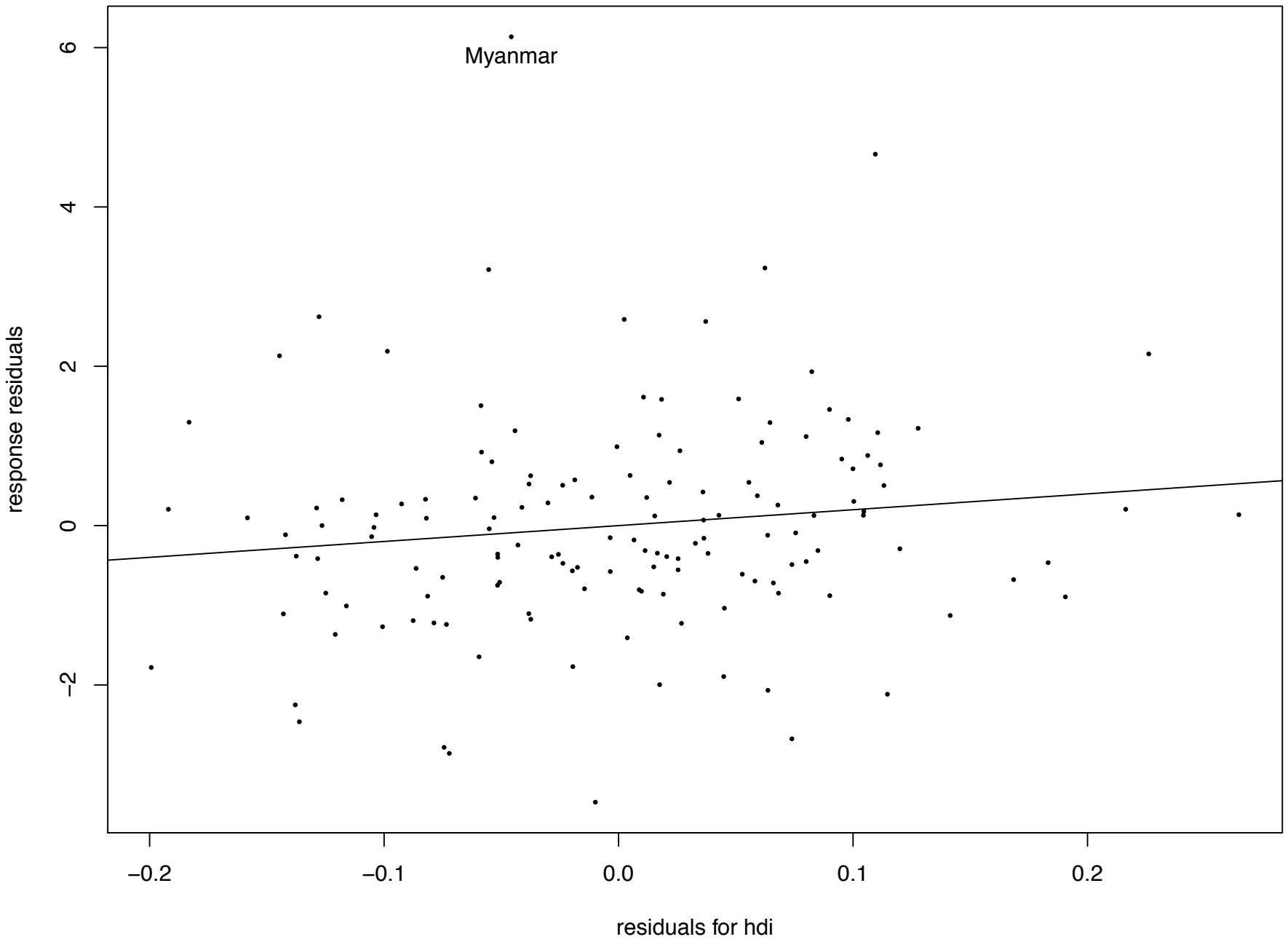
As a result, people recommend this plot to examine the significance of $\widehat{\beta}_j$ in the full regression, examine the data for extreme our outlying points and (to a lesser extent) examine whether a nonlinear function of m_j might improve the fit

An example

Consider the vulnerability data and the effect of HDI -- Let's start with just a simple linear model in all the predictors and examine the added variable plots for a few of them

Again, here we will look at the logarithm of the number of people killed per million in population as the response...

```
fit <- lm(ln_death_risk~ln_events+ln_fert+ln_pop+hdi,data=vul)
fit1 <- lm(ln_death_risk~ln_events+ln_fert+ln_pop,data=vul)
fit2 <- lm(hdi~ln_events+ln_fert+ln_pop,data=vul)
fit3 <- lm(residuals(fit1)~residuals(fit2)-1)

plot(residuals(fit2),residuals(fit1),pch=20,cex=0.5)
abline(fit3)

# coefficients from the full model...

coefficients(fit)

# (Intercept)   ln_events     ln_fert      ln_pop         hdi
#  -5.3484985   1.3708219   2.1960509  -0.5672164   1.9921835

# and from the (through the origin) fit on the residuals
# from the two regressions

coefficients(fit3)

# residuals(fit2)
#        1.992184

# identify points that seem large on the plot...

text(residuals(fit2)[residuals(fit1)>5],
     residuals(fit1)[residuals(fit1)>5],
     vul$country[residuals(fit1)>5],pos=1)
```


Myanmar
response residuals
residuals for hdi
6
4
2
0
−2
−0.2
−0.1
0.0
0.1
0.2

Added variable plots

Suppose we have fit a model and want to include a new variable -- Part of the motivation for the added variable plot construction is the fact that simple residual plots can portray an incorrect image of the dependence of a new variable not yet in the model

If the response is related linearly, say, to a new variable, then plotting the residuals from the current fit against this new variable need not be linear -- It is only guaranteed to be so if the new variable is orthogonal to those already in the model

How could you demonstrate that?

Added variable plots

Added variable plots, on the other hand, correct this situation and if the response depends linearly on the new variable, we see a linear relationship in the plot

However, you can also show that just because this plot is linear, it's not necessarily the case that the underlying relationship is linear -- Correlations with the column space of the variables already in the model are the problem

Another diagnostic

The partial residual plot is an alternative to the standard residual plot we used the other day and is often used to audition new variables not yet in a model -- Let $\widehat{\beta}$ be the least squares fit to our response y using all the variables (both old and the new candidate)

Let $\widehat{\epsilon}$ denote the residuals from this full fit and plot

$$\widehat{\epsilon} + \mathrm{m_j}\widehat{\beta_j} \quad \text{against} \quad \mathrm{m_j}$$

Where does this come from?

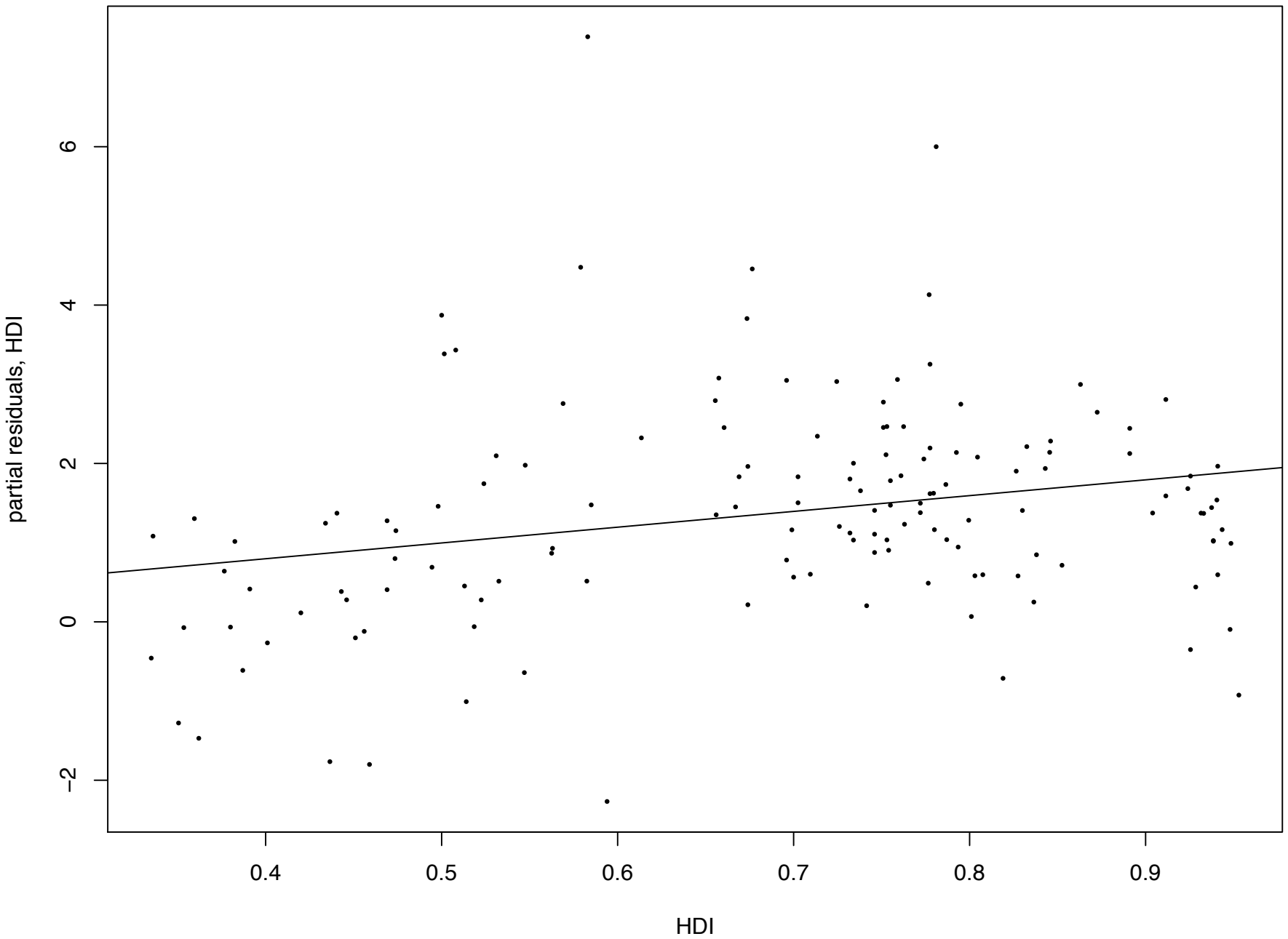
Partial residual plots

To isolate the effect of a single variable m_j, we might consider looking at the response with the effects of the other inputs removed

$$y - \sum_{l \neq j} m_l \widehat{\beta}_l = \widehat{\mu} + \widehat{\epsilon} - \sum_{l \neq j} m_l \widehat{\beta}_l = m_j \widehat{\beta}_j + \widehat{\epsilon}$$

What properties does this plot have?

```
# fit the full model

fit <- lm(ln_death_risk~ln_events+ln_fert+ln_pop+hdi,data=vul)

# extract our model matrix (MMMMMMM)

M <- model.matrix(fit)

# it's really a matrix and its column names are just the variable names in the model

colnames(M)

# [1] "(Intercept)" "ln_events"   "ln_fert"     "ln_pop"      "hdi"

# extract just the HDI column

m <- M[,"hdi"]

# and make the partial residual plot...

plot(m, coefficients(fit)["hdi"]* m + residuals(fit),
            xlab = "HDI",ylab="partial residuals, HDI")

abline(0, coefficients(fit)["hdi"])
```


partial residuals, HDI
6
4
2
0
−2
0.4
0.5
0.6
0.7
0.8
0.9
HDI

Comparisons

In general, ordinary residual plots, added variable plots and partial residual plots are all tools we might consult to examine deficiencies in a model or to audition model elaborations

It is believed that (and our single example supports it) that the partial residual plot is better at spotting nonlinearities (like the need for a quadratic term, say), while the added variable plot is better at understanding outliers and extreme points that may influence the fit