Why validation in Principle Components Analysis?

A Climate Reconstruction Case

Jan van Rongen

8 november 2015

1. Introduction

McIntyre and McKitrick [M&M03, M&M05a, M&M05b] have the critisized methods used by Mann e.a. in [MBH98]. One of their points was that the shape of the global temperature reconstruction in [MBH98] was caused by the method used, not by the underlying data. That evidence is illustrated by figure 3 of [M&M05a], which wants to show that a peculiar data normalisation was the cause because other ways of normalising the data does not give the illustrated "hockeystick" shape.

In this document we replicate that figure 3 from the data published by M&M; in the replication we show where their technique fails or could be substantially improved. In either case we suggest that their misunderstanding of the PCA-method has caused their conclusion to be wrong. In fact we show that [M&M05a]'s proposed alternative normalisation method leads to the same shape and reconstruction if the PCA method is properly applied.

2. Methods used in this document

[M&M05] does not replicate [MBH98] directly (for that see [WA2007]). Instead they simplify the data and work from there, trying to immitate the MBH -processing. As we cannot directly compare their work to the original, we have set ourself the goal to directly replicate figure 3 in [M&M05a].

Before that we will show in a full example with the same data but different normalisation how to perform a reconstruction with Principle Components Analysis form the provided data. This serves to illustrate the method, but also the roles of the PC's in the reconstruction: that they are a tool in the reconstruction, but not the reconstruction itself.

This example fills the second part, then in part 3 we build the replication of figure 3 in [M&M05]. That replication is succesful. While doing so we unearth the errors in the original.

2.1 The technical details

We use the input data from [M&M05a], which is still available on the website of the publisher (http://onlinelibrary.wiley.com/doi/10.1029/2004GL021750/full) as supporting information. The file 2004GL021750-NOAMER.1400.txt there contains 70 tree-ring proxy series collated by M&M from the total of 112 proxy series that were used in [MBH98].

The standard procedure would be:

1. normalize the data
2. select a reference temperature set
3. select the amount of Principal Components needed as follows:

- split the avalailable data in a training (=calibration) and validation set
- try various options out on the training set, resulting in a model that …
- … is used to "predict"" the validation set

- choose the model that performs best in the validation

4. recalculate the selected model on the whole reference set and ...
5. use that to "predict" the period for which no temperature is known.

M&M05a does not follow these steps, it just assumes that the original Principle Components have to be used. We'll come back to that. In any case, there is no reference temperature set in the supporting information, so we have to get that from the site of P.Huybers (who also wrote a critique on the [M&M05a] paper) at http://www.people.fas.harvard.edu/~phuybers/Hockey/ - the file instrumental_sparse was used in [MBH98] and [M&M05a] according to Huybers [Huy2005] and comes from that site.

2.2 The text book example

Here I go through all the steps for a completely standard approach as sketched above. The data is scaled completely which is usually done in practice because in that case the first and second Principal Components have some nice properties. Google - it has been described many times [ST1998] - has built a succesfull empire on the nice properties of the 2nd PC.

Let's get the data and prepare it.

For the training set we use the same period as [MBH98]. As we are dealing with correlated data in time series, we cannot just make a random subselection. The split in two segments is then a logical choice. There are more advanced techniques such as the block bootstrap, see f.i Mudelsee's book [Mud2010].

```
# Input data stored in an R loadable data file;
# otherwise load from above urls
load("./data.Rdata")
# split the instrumental record
y.time <- instrumental_sparse$Year
y <- instrumental_sparse$SPARSE_GRID_INSTR
y.train <- y[y.time >= 1902 & y.time <=1980]
y.val <-  y[y.time >= 1854 & y.time <=1901]
# normalize proxy data on both means and sd (textbook approach)
pHuy<- proxy[,3:72]
pHuy<- sapply(1:ncol(pHuy), function(i)
  return((pHuy[,i]-mean(pHuy[,i]))/sd(pHuy[, i])))

# see the input so it can be checked against the source
(proxy[1:4, 1:8]); (instrumental_sparse[1:4, ]); (dim(proxy)); (dim(instrumental_sparse))
```

```
##   nr year ar049.txt ar050.txt ar052.txt ar053.txt az082.txt az086.txt
## 1  1 1400     0.602     0.019     1.082     0.563     0.769     0.245
## 2  2 1401     1.104     0.552     1.563     0.676     0.710     0.617
## 3  3 1402     1.647     0.713     1.589     1.059     0.746     0.489
## 4  4 1403     1.592     1.144     0.986     1.223     0.876     0.628

##   Year SPARSE_GRID_INSTR SPARSE_GRID
## 1 1854       -0.04389591  -0.2867116
## 2 1855       -0.38537440  -0.3981778
## 3 1856       -0.39890009  -0.2344713
## 4 1857       -0.35755486  -0.1369313

## [1] 581  72

## [1] 140   3
```

2.3 The train/validate cycle We make a sequence of models using the training set, then validate those models on the validationset. We need a validation function, and the standard function almost always used here would be the RMSE, but MBH98 uses RE. Not a problem, just note that we want to minimize RMSE but maximize RE. These are the functions we are going to use:

```r
RMSE<- function(actuals, prediction){
  return(sqrt(mean((actuals-prediction)^2)))
}

RE<- function(actuals, prediction){
  return(1-sum((actuals-prediction)^2)/sum(actuals^2))
}

# The following function trains and validates a single case with N principle components

case_train <- function (svd0, train, val, y.train, y.val, N) {
  # svd0: svd for the whole set
  # train: subset of proxy for training years
  # val: subset of proxy for validation years
  # y.train, y.val: as above
  #
  # for convenience use the alternative way to calculate the PC's
  x = train %*% svd0$v[,1:N]
  l = lm(y.train ~ x) # linear model makes the coefficients for the PC's
  x = val %*% svd0$v[,1:N]
  yhat = predict(l, newdata = as.data.frame(x))
  val.value = RE(y.val, yhat)
  train.value = RE(y.train, l$fit)
  val2.value = RMSE(y.val, yhat)
  train2.value = RMSE(y.train, l$fit)
  return(c(REtrain=train.value, REval=val.value, RMSEtrain=train2.value, RMSEval=val2.value))
}
```

We are now ready to find the "best set" of Principle components. We will try the lower half of all cases

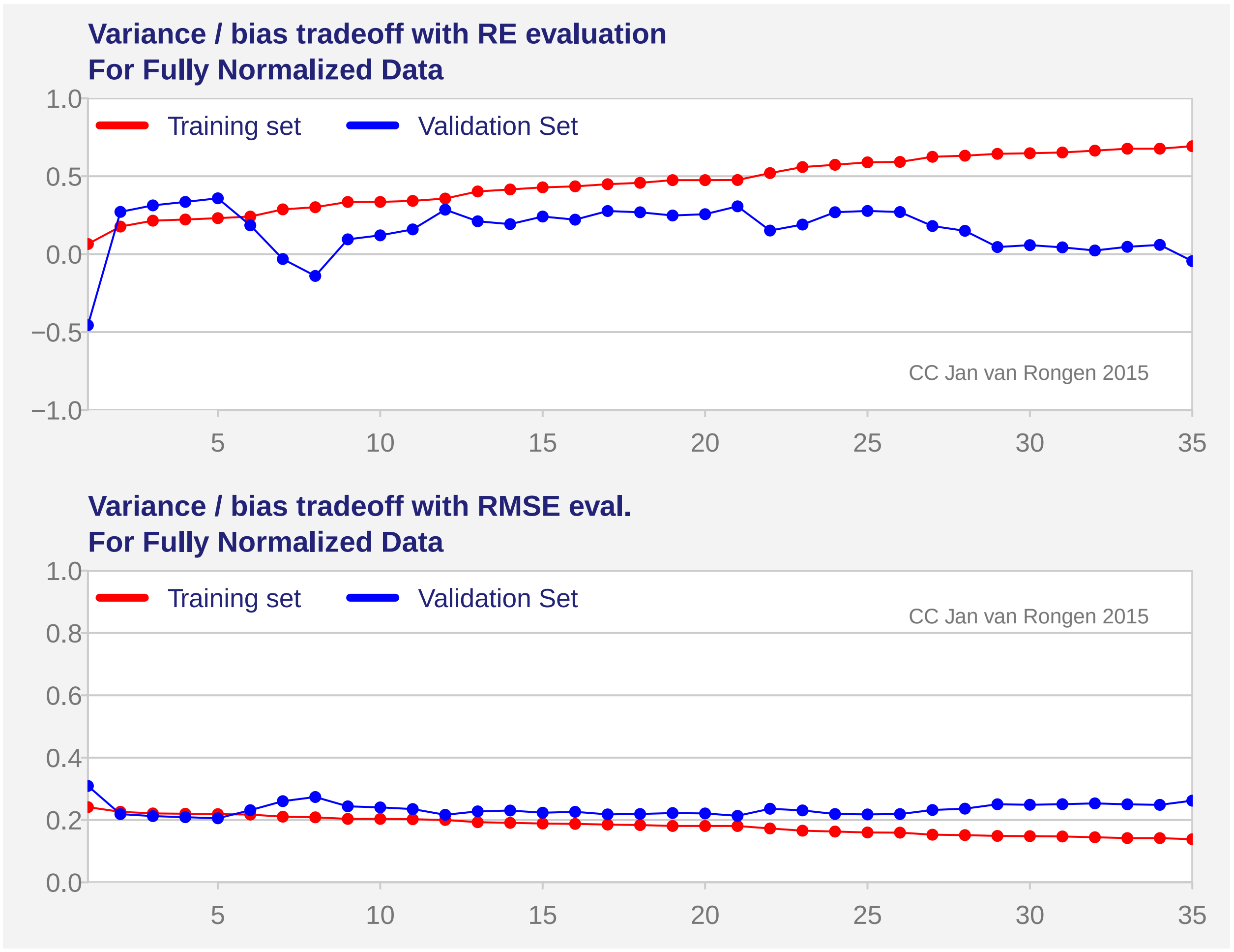
```r
valHuy<-pHuy[proxy$year >= 1854 & proxy$year <= 1901,]
trainHuy<-pHuy[proxy$year>= 1902,]
# The svd decomposes Z = u.d.t(v) with u the principle components, and also u=Z.v.(1/d)
svdHuy=svd(pHuy)
result<-sapply(1:35, function(N){
  return(case_train(svdHuy, trainHuy, valHuy, y.train, y.val, N))
})

(result[2,5])
```

```
##     REval
## 0.3583554
```

Plot the results of all the cases with two different evaluation functions.

```r
op<-par(no.readonly = TRUE) # save old plot parameters
library(myLib) # contains enhanced plotting routines
```

```
par(mfrow=c(2,1))

pretty_plot(ts(result[1,]), ylim=c(-1, 1)
            , main="Variance / bias tradeoff with RE evaluation\nFor Fully Normalized Data"
            , kleur=1)
pretty_plot(ts(result[1,]), add=T, type="p", kleur=1, cex=0.7)
pretty_plot(ts(result[2,]), add=T, type="p", kleur=2, cex=0.7)
pretty_plot(ts(result[2,]), add=T, kleur=2)
pretty_legend(lwd=4, kleur=c(1,2), c("Training set", "Validation Set"))


pretty_plot(ts(result[3,]), ylim=c(0, 1)
            , main="Variance / bias tradeoff with RMSE eval. \nFor Fully Normalized Data"
            , kleur=1, ccloc=3)
pretty_plot(ts(result[3,]), add=T, type="p", kleur=1, cex=0.7)
pretty_plot(ts(result[4,]), add=T, type="p", kleur=2, cex=0.7)
pretty_plot(ts(result[4,]), add=T, kleur=2)
pretty_legend(lwd=4, kleur=c(1,2), c("Training set", "Validation Set"))
```



So we see that the fit to the training set of proxy data becomes better while we add more PC's to the linear model, but the bias of "overfitting" steps into the game after 5 PC's. The real art (or science) of Statistical Learning is to find good algorithms that help strike the balance between the bias of overfitting and the noise of too much variance. By the way we see that (in this case) RE is as good an evaluation fucntion as the common RMSE.

2.4 Reconstruction = predicitng the past

Now that 5 has been established as the proper number of PC's we will make a complete prediction. So we will apply our principle components-model to the complete y and then reconstruct the rest.

```
y.instrumental<-c(y.val, y.train)
x<-svdHuy$u[,1:5]; x<- x[proxy$year >= 1854 & proxy$year <= 1980,]
l= lm(y.instrumental ~x)
x<-svdHuy$u[,1:5]
pred<-predict(l, newdata=as.data.frame(x)) # equivalent to x %*% coef(l)[2:6]
# These are coefficients in the linear model
(coef(l))
```

```
## (Intercept)           x1           x2           x3           x4           x5
##  -0.1965343   0.4868999   3.0332635  -1.0001238  -0.5064829  -0.4019323
```

2.5 A plot of results and contribution of the PC's

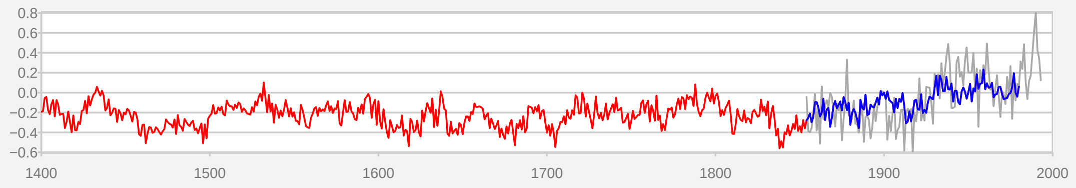
We have not plotted the result yet, because we want to draw the attention to the fact that PC's do not necessarily contribute positively to the result. The coefficients of the linear model can be negative f.i. Not all PC's contribute a great extend to the prediction.

There is a simple explanation for that. The PC's explain the maximum variance, and the more variance we leave in the original data the less the (first) PC's are able to see the underlying trend.

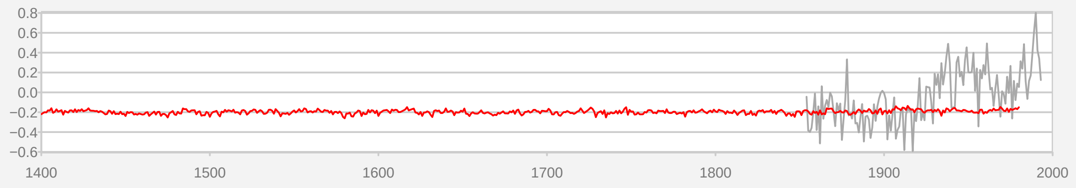
So I am going to multiply all these PC's by their contribution to the model and add the intercept, so that the line up well against the background of the instrumental signal. You will notice that the PC that contributes most (PC2) was already predicted in the train/validation cycle. But what you couldn't see there is that PC3-5 have a slightly negative slope.

```
PC_to_plot<- l$coef[1] + sapply(1:5, function(i) return(svdHuy$u[,i]*l$coef[i+1]))

par(op) # reset plotting params
par(mfcol=c(6,1))
pretty_plot(ts(y, start=1854, freq=1), kleur=4
            , xlim=c(1400, 2000)
            , main="Reconstruction using Fully normalized data and 5 PC's", ccloc=0)
pretty_plot(ts(pred, start=1400, freq=1), kleur=1, add=T)
pretty_plot(ts(l$fit, start=1854, freq=1), kleur=2, add=T)
for (i in 1:5){
  pretty_plot(ts(y, start=1854, freq=1), kleur=4
            , xlim=c(1400, 2000)
            , main=sprintf("Contribution of PC %d", i), ccloc=0)
  pretty_plot(ts(PC_to_plot[, i], start=1400, freq=1), kleur=1, add=T)

}
```

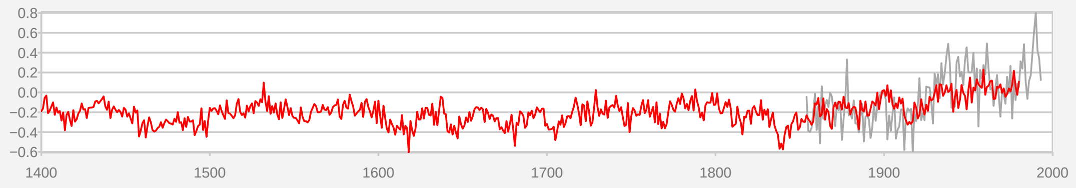
Reconstruction using Fully normalized data and 5 PC's



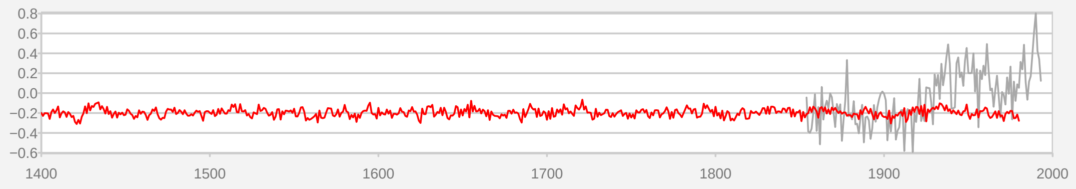
Contribution of PC 1



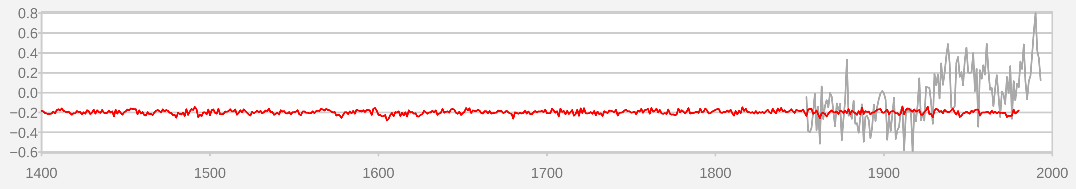
Contribution of PC 2



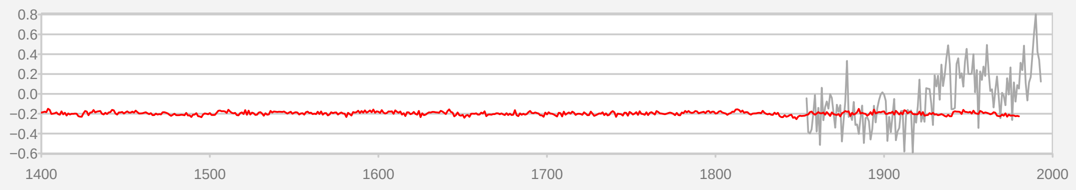
Contribution of PC 3



Contribution of PC 4



Contribution of PC 5



2.6 What we have learned from this example.

Once you see how it is done, it is pretty simple, isn't it? There's not much room for error.

We have seen the need to select the number of relevant PC's, based on the data and a known example result. We have seen that PC's do contribute different amounts to the reconstruction, sometimes negative. We have seen that most PC's do not even follow the shape of the trend at all.

Well, MBH98 had much more complex data to work with than we used here. Their dimensions were space and time. M&M03/05 simplified it to time series only set with 70 proxies. So we cannot rely on the MBH98's way of selecting the principle components, we need to redo the selection with this data. MBH98 used a method attributed to Preisendorfer (called rule N - available in the R package `wq`). We could have repeated that here, but it is complex and leads to very similar results. For this simplified data set our method seems robust. So I won't bother investigating others here.

It's time to replicate that figure 3.

3. Replication

M&M shifted the data with the mean of the proxy columns, but did not scale them with the column standard deviation. Only one line difference with the code for full scaling, but what a difference does it make in the number of PC's that is best for the prediction with this type of model. . .

```
pMM<- proxy[,3:72]
pMM<- sapply(1:ncol(pMM), function(i){
  return(pMM[,i]-mean(pMM[,i]))}
  )
valMM<-pMM[proxy$year >= 1854 & proxy$year <= 1901,]
trainMM<-pMM[proxy$year>= 1902,]
# The svd decomposes Z = u.d.t(v) with u the principle components, and also u=Z.v.(1/d)
svdMM=svd(pMM)
resultMM<-sapply(1:70, function(N){
  return(case_train(svdMM, trainMM, valMM, y.train, y.val, N))
})

(which(resultMM[2,]==max(resultMM[2, ])))
```

```
## [1] 12
```

So the M&M method needs 12 PC's for the best amongst all the predictions that this input set can give with PC's, and 9-11 are nearly as good. The evaluation on the validation is quite good, better than the previous example.

But in reality they only used 2 PC's, and the RE value for only 2 PC's is negative. That means you model the noise, not the trend.

```
(resultMM[2,1:8]); (resultMM[2, 9:12])
```

```
## [1] -0.1845266083 -0.4256347166 -0.3837657788  0.0472288840  0.1623819118
## [6] -0.0225074397  0.0003632705  0.0881223732

## [1] 0.4312877 0.4322292 0.4329599 0.4334409
```

Nine will have less bias, so I would probably go for that. Also note that the results make a big jump forward with PC9, not with the earlier ones.

3.1 Data for MBH98 uses normalisation plus detrending

Now we come to the difficult part. MBH rescale the data for the period 1902-1980, which makes sense *in this case* for the training/validation, but not for the prediction. On the other hand, as we have seen in the above examples, you might get a better result with a non standard approach. Then MBH uses a second round of data conversion: detrending using the temperature-data from 1902-1980, Again this might be very useful in the training/validation, but it certainly does not help to understand what is happening.

To make things even more complicated: not everybody tells the same story. Huybers says that it is slightly different, and Ammanns code (WA07) in fact rescales it back later, but let's just skip that for now and follow M&M because we want to make clear what they did.

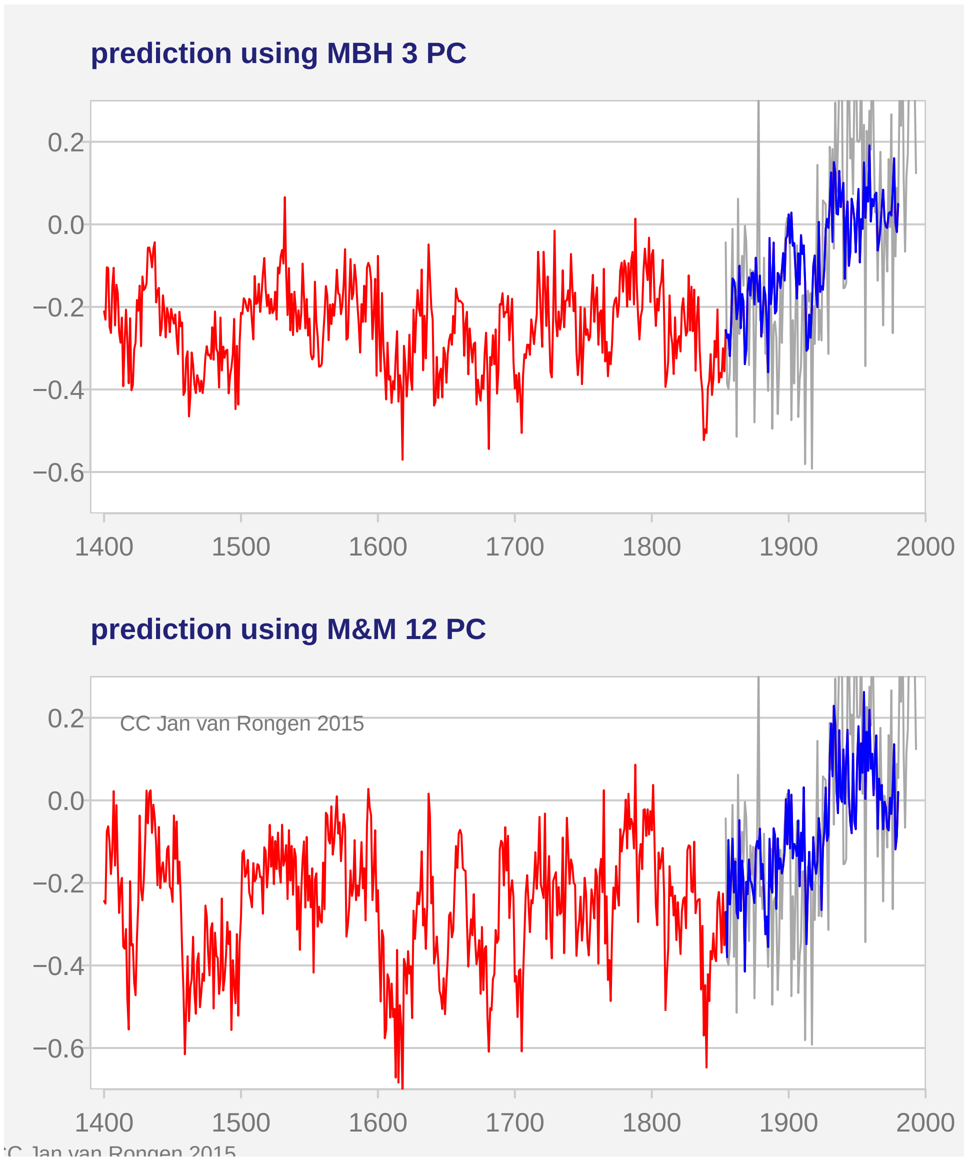
```r
# detrend removes the linear trend in any vector
detrend<-function(y){
  y<-as.vector(y); x<-1:length(y)
  return(lm(y~x)$res)
}

# the MBH way
pMBH<-proxy[,3:72]

pMBH.anch<-pMBH[proxy$year >= 1902,] # anchors the scaling here
m.anch<-apply(pMBH.anch,2,mean)
s.anch<-apply(pMBH.anch,2,sd)
pMBH<-scale(pMBH,center=m.anch ,scale=s.anch)     # that part 1
sdprox<-sd(detrend(pMBH[proxy$year >= 1902,]))
pMBH<-scale(pMBH,center=FALSE,scale=rep(sdprox,70 ))
valMBH<-pMBH[proxy$year >= 1854 & proxy$year <= 1901,]
trainMBH<-pMBH[proxy$year>= 1902,]
svdMBH=svd(pMBH)
resultMBH<-sapply(1:70, function(N){
  return(case_train(svdMBH, trainMBH, valMBH, y.train, y.val, N))
})

(which(resultMBH[2,]==max(resultMBH[2, ]))); (resultMBH[2,1:5])
```

```
## [1] 3

## [1] 0.2102859 0.3547349 0.3660905 0.3505610 0.3265437
```

So, it's three, with RE score slightly above full scaling, but below M&M. But 2 also looks quite good, it will have less bias, so I would probably go for that. Note where the big jump is.

3.2 M&M proper would also have predicted a "hockey-stick"

Without repeating the code: show the results of the optimal predictions for both cases.



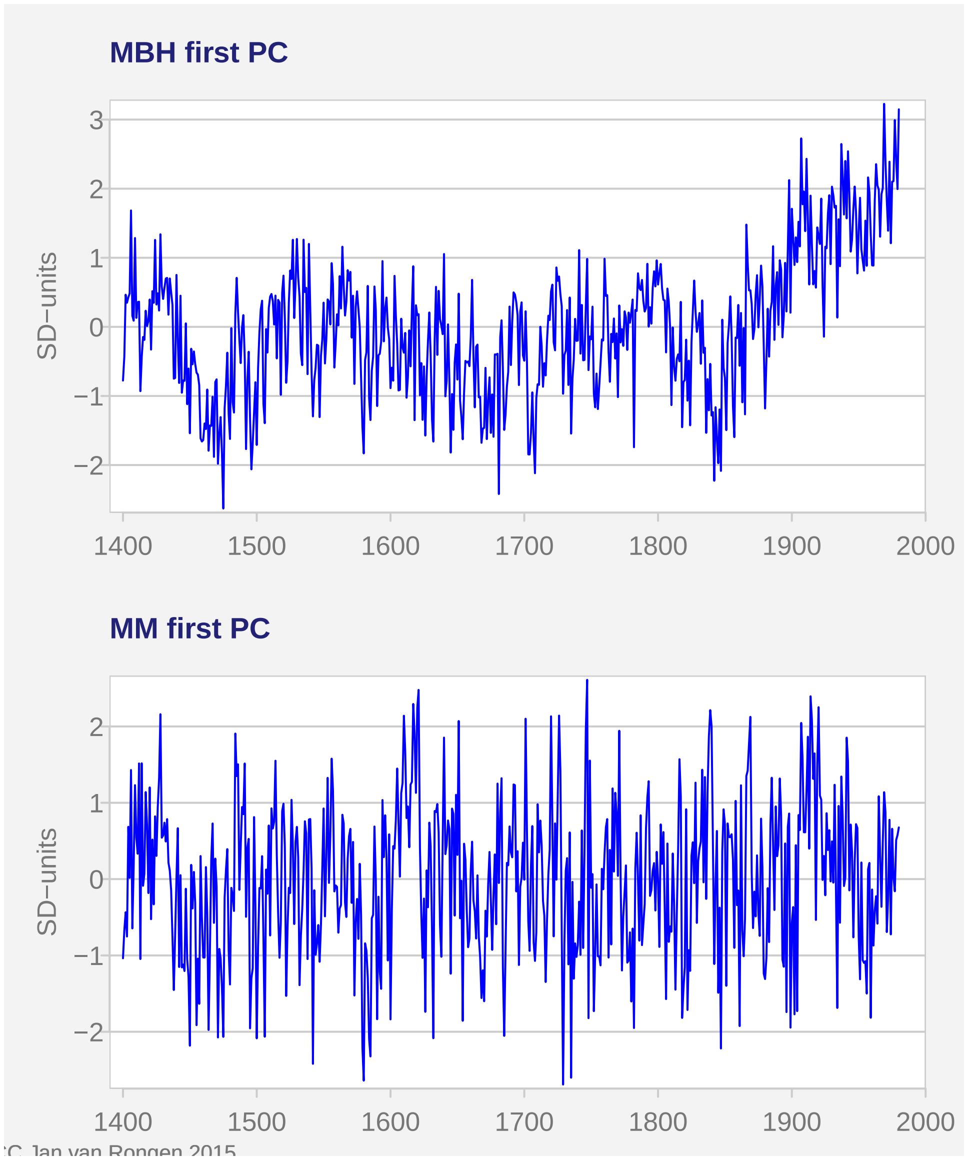
3.3 The reconstructed figure 3

Figure 3 however only shows the first principle components, and those components have been normalised by M&M before they are shown. So there are three misleading suggestion in that figure: 1. it suggests that the first PC shows (part of) the trend of the reconstruction, which is wrong 2. it suggest that the upper and lower half of the figure show the PC's at the same scale, which is also wrong. 3. the footnote says they have been rescaled to the 1902-1980 period, which is wrong because the vertical axis is not a temperature but still a standard deviation unit. Rescaling would have implied the same vertical scale. See paragraph 4 for the really rescaled version.

My reconstruction:

```
# remember we still have the data
#
par(op)
par(mfrow=c(2,1))
y.instrumental<-c(y.val, y.train)
```

```
PC1.MBH <- pMBH %*% svdMBH$v[,1]
PC1.MBH <- (PC1.MBH-mean(PC1.MBH))/sd(PC1.MBH)
pretty_plot(ts(PC1.MBH, start=1400, freq=1), kleur=2
            , xlim=c(1390, 2000), ccloc=0, ylab="SD-units"
            , main="MBH first PC")

PC1.MM <- pMM %*% svdMM$v[,1]
PC1.MM <- (PC1.MM-mean(PC1.MM))/sd(PC1.MM)
pretty_plot(ts(PC1.MM, start=1400, freq=1), kleur=2
            , xlim=c(1390, 2000), ccloc=0, ylab="SD-units"
            , main="MM first PC")
```



3.4 Compare

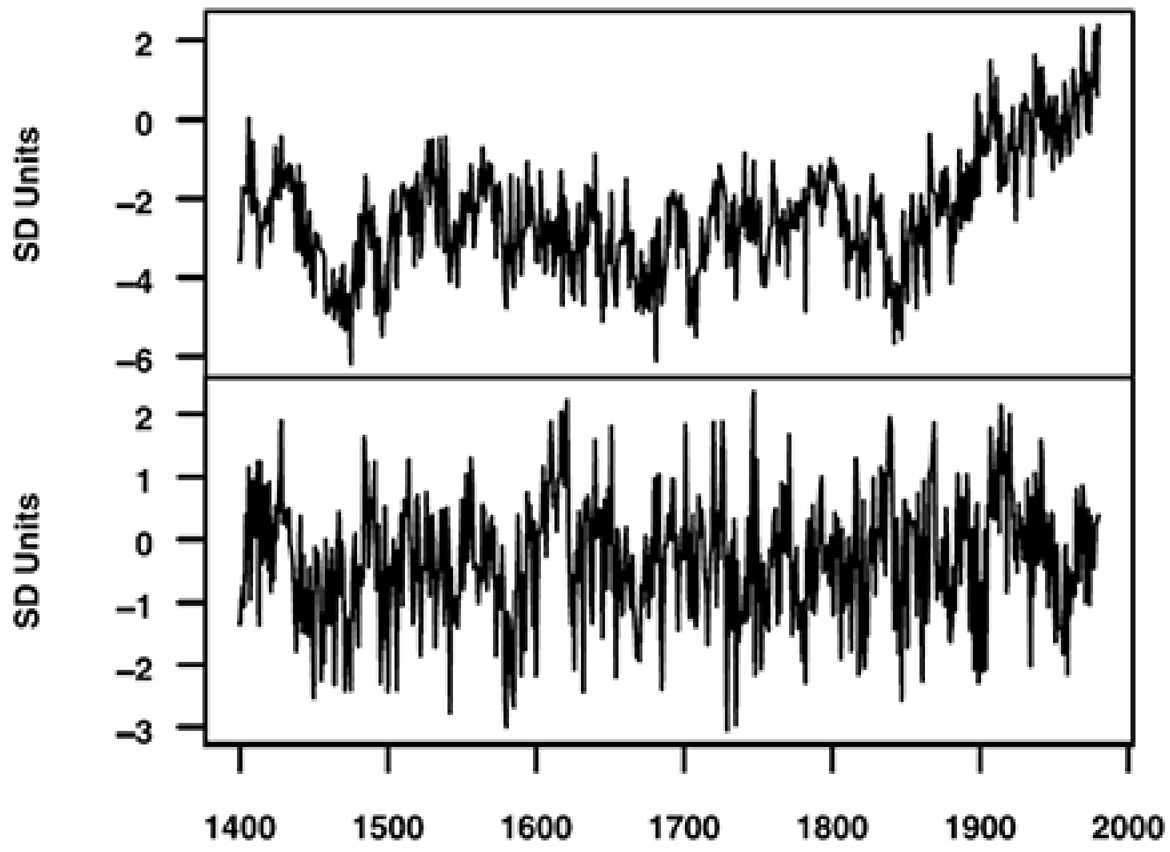
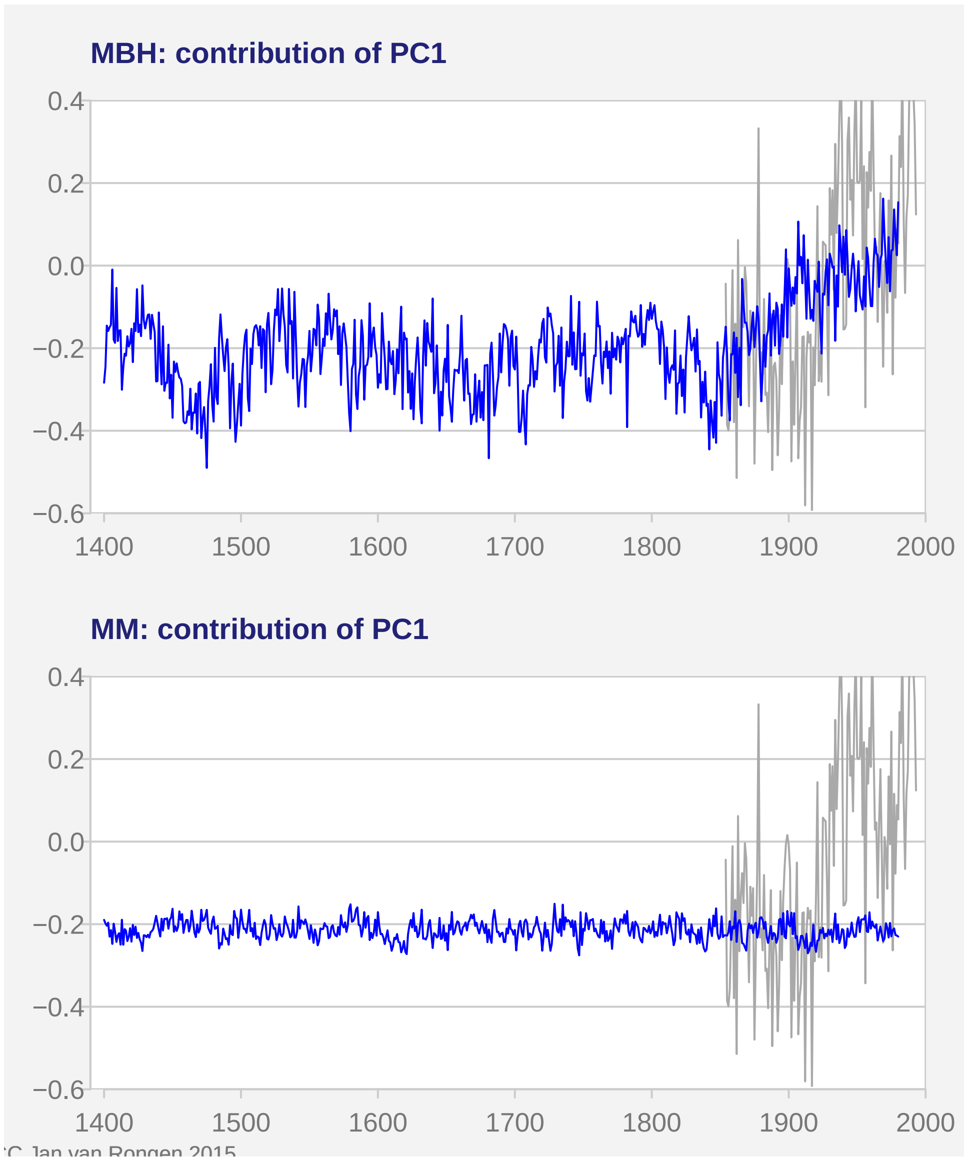


Figure 3. PC1 for AD1400 North American Tree Ring Network. Top: Result with MBH98 data transformation; Bottom: recalculated on the same data without MBH98 data transformation. Both standardized to 1902–1980 period.

Yes, that looks exactly the same, except for the vertical scales. There is something strange there: my SD-units for the first part are different. Let's forget it. The whole figure is already wrong: we have shown that this figure compares the wrong PC's and that in any case the existence of a trend in the PC does not always mean that the PC contributes much to the overall prediction. This M&M first PC just shows the variation that was not normalised out beforehand. In order to pick up a useful signal, at least 9 PC's should have been used and at most 12.

4. Conclusion

We have shown that the figure 3 of M&M05 is misleading or even wrong, because it tries to tell the untrue story that the first Principle Component is always the most dominant one. What should have been shown is the contribution of those PC's to the overall prediction. I will show that figure now, together with the instrumental data (as shown before). If you look at this figure it tells a completely different story. In fact, if this had been figure 3, it would have looked quite silly in its context:



M&M should have used a proper PC-selection plus validation - and would have found the hockey-stick also with their version of normalisation .

References

- ESL2013 - Elements of Statistical Learning, 2nd ed., 2013, by T. Hastie, R. Tibshirani & J. Friedman. Free copy available online on Hastie's site.
- Mud2010 - Mudelsee: 2010. Climate Time Series Analysis, Classical Statistical and Bootstrap Methods. Springer Verlag.
- MBH98 - Mann, M. E., Bradley, R. S., and Hughes, M. K.: 1998, 'Global-scale temperature patterns and climate forcing over the past six centuries', Nature 392, 779-787.
- M&M03 - McIntyre, S. and McKitrick, R.: 2003, 'Corrections to the Mann et al (1998) proxy data base and Northern Hemispheric average temperature series', Energy and Environment 14, 751-771.
- M&M05a - McIntyre, S. and McKitrick, R.: 2005, 'Hockey sticks, principal components, and spurious significance', Geophys. Res. Lett. 32, L03710.

- M&M05b - McIntyre, S. and McKitrick, R.: 2005, 'The M&M critique of the MBH98 Northern Hemisphere climate index: update and implications', Energy and Environment 16, 69-100.
- Huybers2005 - Huybers, P.:2005, 'Comment on "Hockey sticks, principal components, and spurious significance" by McIntyre and McKitrick', Geophys. Res. Lett 32, L20705.
- WA2007 - Wahl, E. R., & Ammann, C. M. (2007). Robustness of the Mann, Bradley, and Hughes reconstruction of northern hemisphere surface temperatures: examination of criticisms based on the nature and processing of proxy climate evidence. Climatic Change, 85, 33–69.
- R - R Core Team (2015). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/
- ST1998 - [L. Page, S. Brin, R. Motwani, and T. Winograd: 1998. The PageRank citation ranking: Bringing order to the web, tech. rep., Stanford Digital Library Technologies Project.