Projected sea level rise in Oslo

Peter Guttorp

September 24, 2015

This documents goes through and explains the code used to project mean sea level rise in Oslo from temperature data, climate models, and station data.

Some default options, loading libraries and data

```
options(stringsAsFactors = FALSE)
rel="data"
rcp="all"
conf=0.90 #default confidence level
gia=0
se.gia=0

plots=1:10
```

To plot, say, figures 1, 3 and 7 one would set plots=c(1,3,7)

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

###
### Load needed libraries
###
require(ncdf)
```

```
## Loading required package: ncdf
```

```
require(forecast)
```

```
## Loading required package: forecast
## Loading required package: zoo
##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: timeDate
## This is forecast 6.1
```

```
require(excursions)
```

```
## Loading required package: excursions
## Loading required package: Matrix
##
## Attaching package: 'Matrix'
##
## The following objects are masked from 'package:base':
##
##     crossprod, tcrossprod
##
## Loading required package: sp
## Loading required package: spam
## Loading required package: grid
## Spam version 1.0-1 (2014-09-09) is loaded.
## Type 'help( Spam)' or 'demo( spam)' for a short introduction
## and overview of this package.
## Help for individual functions is also obtained by adding the
## suffix '.spam' to the function name, e.g. 'help( chol.spam)'.
##
## Attaching package: 'spam'
##
## The following objects are masked from 'package:base':
##
##     backsolve, forwardsolve
```

```
require(chron)
```

```
## Loading required package: chron
```

```
###
###function to compute covariance structure for fitted ARMA model
###
ARMAcov=function(AR,n)
{
    if (AR$arma[6]>0)
        {print("Error: Integrated model")
        return}
        ar=AR$arma[1]
        ma=AR$arma[2]
        order=ar+ma
        if(order==0) {
            cov=AR$sigma2*c(1,rep(0,(n-1)))
            }
        else if(ar==0)
            {
            cov=AR$sigma2*ARMAacf(0,AR$coef[1:ma],lag.max=(n-1))
            }
        else
            {
                cov=AR$sigma2*ARMAacf(AR$coef[1:ar],AR$coef[(ar+1):order], lag.max
=(n-1))
                }
        return(cov)
        }


###
### Download global data
###
```

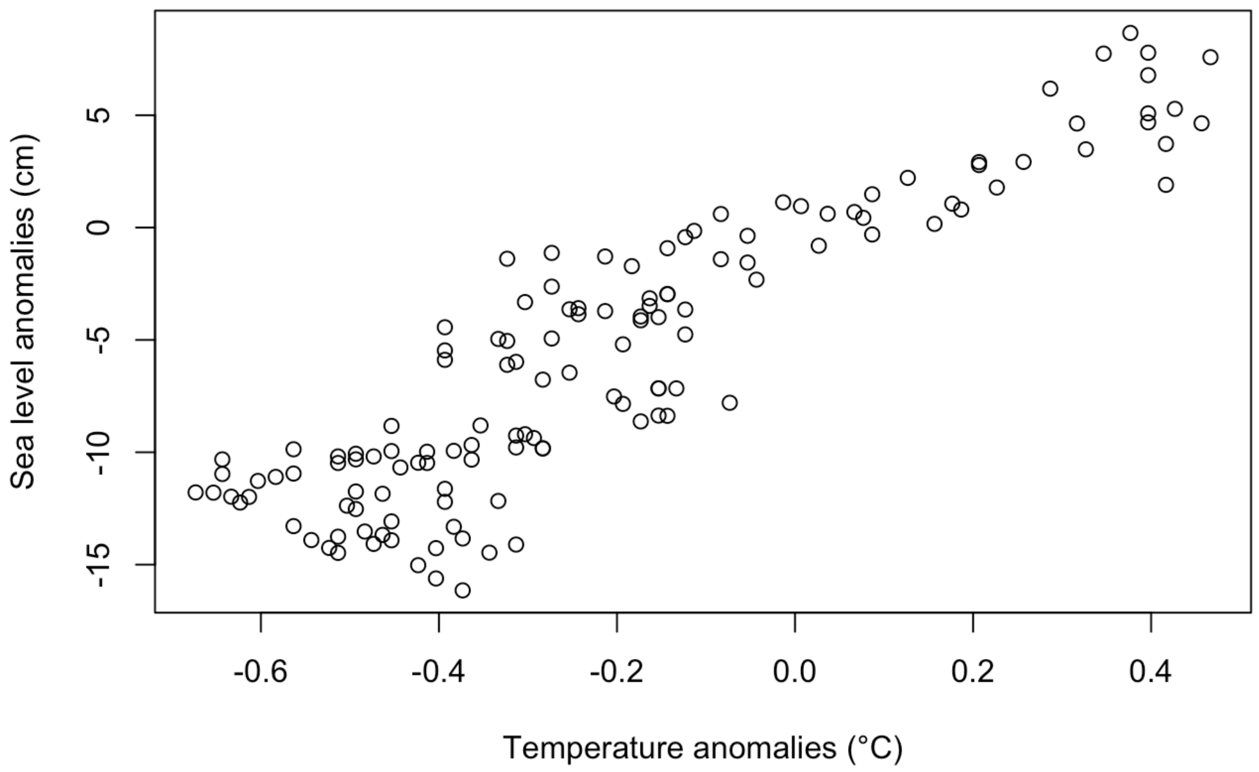
The read.table() calls will be replaced by WP1 code

```
temp = read.table("~/Dropbox/Sea level projections/giss.txt",na.strings="*") #1880
-2014
temp.years=temp[,1]
temp = temp[,2]
church.sealevel = read.table("~/Dropbox/Sea level projections/church_white_gmsl_20
11_up/CSIRO_Recons_gmsl_yr_2015.txt") #1880-2013 monthly
sea.years =floor(church.sealevel[,1])
sd1999 = church.sealevel[which(is.element(sea.years,1999)),3]/10 #put into cm
sealevel = church.sealevel[,2]/10 #put into cm

#
#center these with respect to 1970-1999 anomalies
#

temp = temp-mean(temp[which(is.element(temp.years,1970:1999))])
sealevel = sealevel-mean(sealevel[which(is.element(sea.years,1970:1999))])
```

Question: Do we want the reference years to be user settable?

Plot 1: Sea level against temperature

```
###

tmp=intersect(sea.years,temp.years)#make the seriescover the same years
sealevel=sealevel[which(is.element(sea.years,tmp))]
temp=temp[which(is.element(sea.years,tmp))]

if(sum(is.element(plots,1))>0) {
    plot(temp,sealevel,xlab="Temperature anomalies (°C)",ylab="Sea level anomalies
 (cm)")
 }
```



Sea level anomalies (cm)
5
0
-5
-10
-15
-0.6
-0.4
-0.2
0.0
0.2
0.4
Temperature anomalies (°C)

```
#get 1999 levels (used for integrating)
s1999 = sealevel[which(is.element(sea.years,1999))]

###
### Load in the climate RCPs
###

download.file("http://www.stat.washington.edu/peter/593/ForPeter/glbtas.cmip5.rcps
.1850-2300.r1i1p1.nc","glbtas.cmip5.rcps.1850-2300.r1i1p1.nc")
#Obtained from Claudia Tebaldi, Climate Central

cmip5=open.ncdf("glbtas.cmip5.rcps.1850-2300.r1i1p1.nc")
cmip5.data=get.var.ncdf(cmip5,"glbtas") #global mean temperatures

#turn 9e36 values into NAs
for(i in 1:451) {for(j in 1:45) {for(k in 1:4) cmip5.data[i,j,k]=ifelse(cmip5.data
[i,j,k]>9e36,NA,cmip5.data[i,j,k])}}

model.years=1850:2300
cmip5.data=cmip5.data-273.15 #Convert from K to °C-

for(i in 1:45) {for (k in 1:4) cmip5.data[,i,k] = cmip5.data[,i,k]-mean(cmip5.data
[which(is.element(model.years,1970:1999))
,i,k],na.rm=T)} #anomalies wrt 1970-1999

#indexes of projection years 2000-2100
proj=which(is.element(model.years,2000:2100))

#pick only models with complete data 2000-2100, should return 18 models
models.all=NULL
for(i in 1:45) if(sum(sum(is.na(cmip5.data[proj,i,])))==0)models.all=c(models.all,
i)


#pick models with historical forcings , should return 38 models
hist.models=NULL
for(i in 1:45) if(sum(sum(is.na(cmip5.data[which(is.element(model.years,1880:1999)
)
,i,2])))==0)hist.models=c(hist.models,i)

#pick specific scenario models, should return respectively 26, 36, 18 and 33 model
s
models2.6=NULL
for(i in 1:45) if(sum(sum(is.na(cmip5.data[proj,i,1])))==0)models2.6=c(models2.6,i
)

models4.5=NULL
```

```
for(i in 1:45) if(sum(sum(is.na(cmip5.data[proj,i,2])))==0)models4.5=c(models4.5,i
)

models6.0=NULL
for(i in 1:45) if(sum(sum(is.na(cmip5.data[proj,i,3])))==0)models6.0=c(models6.0,i
)

models8.5=NULL
for(i in 1:45) if(sum(sum(is.na(cmip5.data[proj,i,4])))==0)models8.5=c(models8.5,i
)

length(models2.6)
```

```
## [1] 26
```

```
length(models4.5)
```

```
## [1] 36
```

```
length(models6.0)
```

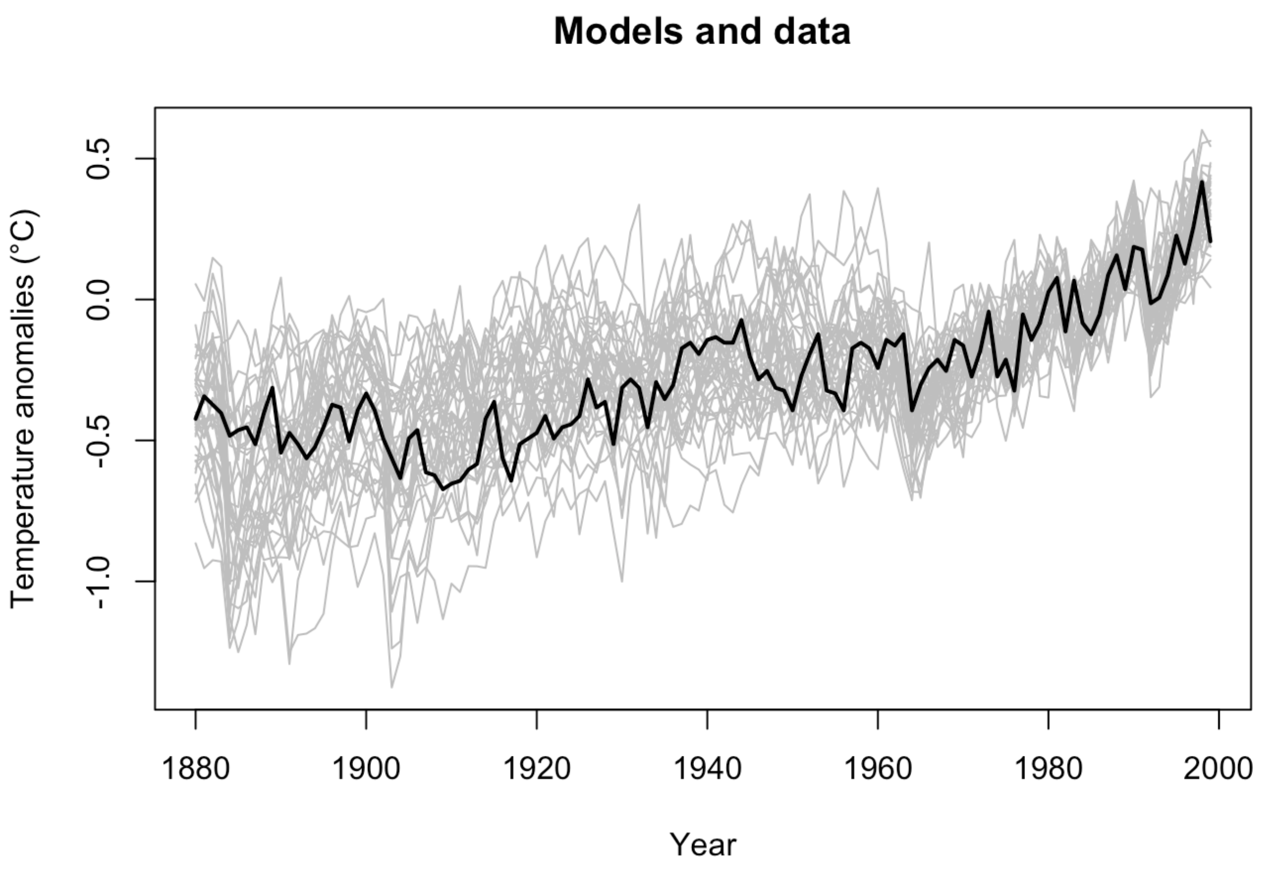
```
## [1] 18
```

```
length(models8.5)
```

```
## [1] 33
```

Plot 2. Historical climate model runs and global mean temperature series

```
if(sum(is.element(plots,2))>0) {
    ylim=range(union(cmip5.data[31:150,hist.models,2],temp[1:120]))
    plot(range(temp.years[1:120]),ylim,xlab="Year",ylab="Temperature anomalies (°C
)",type="n",main="Models and data")
    for(i in hist.models) lines(temp.years[which(is.element(temp.years,1880:1999))
], cmip5.data[which(is.element(model.years,1880:1999)),i,2],col="grey")
    lines(temp.years[which(is.element(temp.years,1880:1999))],temp[which(is.elemen
t(temp.years,1880:1999))],lwd=2)
}
```



Historical global relationship

We can now either estimate the historical relationship between global mean sea level change and global mean temperature using historical temperature data, or using historical model runs (not yet implemented in this code).

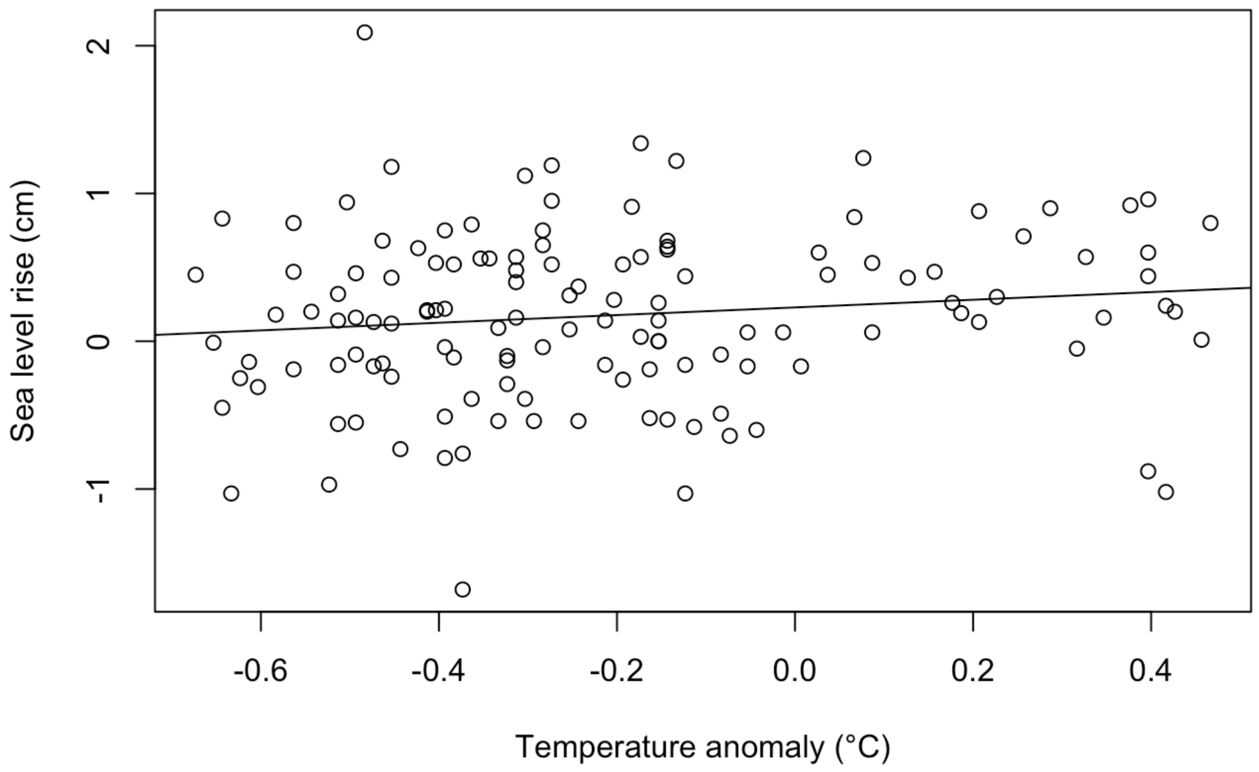
```
if(rel=="data"){

###
### Estimating global relationship from data
###

rate.sea = diff(sealevel)

#the differencing procedure leaves us with one fewer observation so:

temp.raw = temp[-1]


armodel = auto.arima(rate.sea,xreg = as.matrix(temp.raw),stepwise=F,approximation=
F)
summary(armodel)
}
```

```
## Series: rate.sea
## ARIMA(0,0,2) with non-zero mean
##
## Coefficients:
##           ma1      ma2  intercept  as.matrix(temp.raw)
##       -0.4676  -0.1787     0.2294               0.2596
## s.e.   0.0835   0.0900     0.0207               0.0595
##
## sigma^2 estimated as 0.2665:  log likelihood=-101.02
## AIC=212.03   AICc=212.5   BIC=226.48
##
## Training set error measures:
##                        ME      RMSE       MAE MPE MAPE      MASE
## Training set -0.001054965 0.5162621 0.3982108 Inf  Inf 0.5505218
##                     ACF1
## Training set 0.01409807
```

```
a1 = armodel$coef[3]
a2 = armodel$coef[4]
```

Plot 3. Global mean temperature against differenced global mean sea level with time series regression line

```
if(sum(is.element(plots,3))>0) {
    plot(temp.raw,rate.sea,xlab="Temperature anomaly (°C)",ylab="Sea level rise (c
m)")
    abline(a1,a2)}
```



```
###
### Download Oslo station series
###

oslo.m= read.table("~/Dropbox/Sea level projections/oslo.monthly.txt",sep=";",na.s
trings="-99999") # From http://www.psmsl.org/data/obtaining/stations/62.php

#Compute annual means
oslo.mean=tapply(oslo.m[,2],floor(oslo.m[,1]),mean,na.rm=T)

#Deal with missing values
sta.years=unique(floor(oslo.m[,1]))
sta.name="Oslo"
sta=oslo.mean
sta=sta/10 #cm
sta.range=which(is.element(sta.years,1916:2013))
sta.years=sta.years[sta.range] #too much missing data before 1916
sta=sta[sta.range]
sta.miss=(1:length(sta))[is.na(sta)] #Find missing value(s)

if(length(sta.miss)>1) stop("Error, too many missing values")
sta[sta.miss]=mean(sta[c(sta.miss-1,sta.miss+1)]) #linear interpolation for 1939 w
hich is missing for all months

sta = sta-mean(sta[which(is.element(sta.years,1970:1999))],na.rm=T) #Anomalies wrt
 1970-1999
```

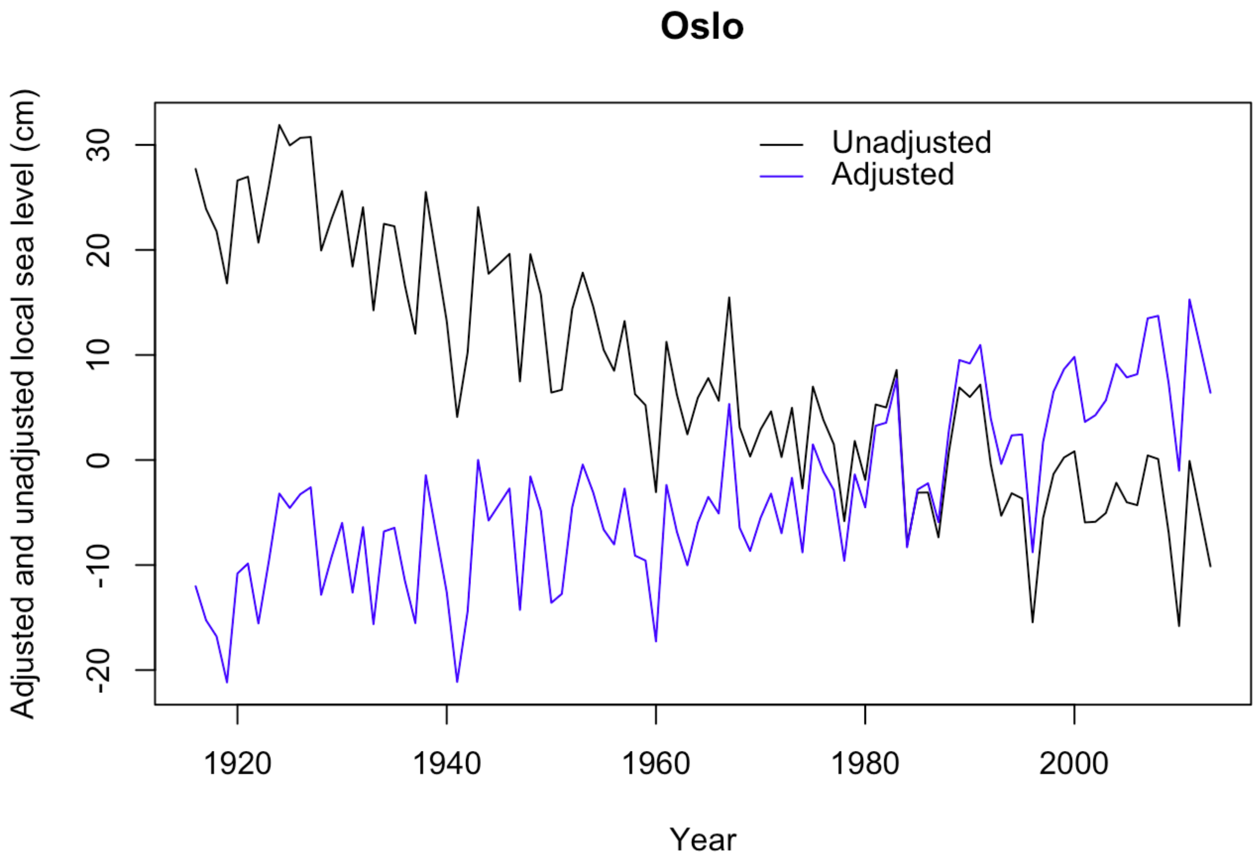
Glacial isostatic adjustment

The ground at Oslo is rising relatively rapidly (in fact faster than the sea level rise) as a rebound from the latest ice age. This needs to be taken into account in the projections.

```
gia=0.58 #cm/yr #from Sullivan et al., 2014
se.gia=0.05#ignoring reference frame uncertainty at the moments

# Common years to use for estimating local relationship

tmp=intersect(sta.years,sea.years)
sea.common.years=which(is.element(sea.years,tmp))
sta.common.years=which(is.element(sta.years,tmp))



N=length(sta.years)-1
reg=0:N
adj=mean(reg[which(is.element(sta.years,1970:1999))])#anomaly reference

reg=reg-adj
sta.un=sta #uncorrected station series
sta=sta+gia*reg #gia correction
reg0=reg[which(is.element(sta.years,2000))]
```

Plot 4 Adjusted and unadjusted local series

```
ytmp=range(sta,sta.un,na.rm=T)
if(sum(is.element(plots,4))>0) {
    plot(range(sta.years),ytmp,xlab="Year",ylab="Adjusted and unadjusted local sea
 level (cm)",type="n",main=sta.name)
    lines(sta.years,sta.un)
    lines(sta.years,sta,col="blue")
    segments(1970,30,1974,30)
    segments(1970,27,1974,27,col="blue")
    text(1975,30,"Unadjusted",pos=4)
    text(1975,27,"Adjusted",pos=4)
    }
```



Estimate relationship with global sea level from gia-corected data

```
ar.sta=auto.arima(sta[sta.common.years],xreg=as.matrix(sealevel[sea.common.years])
,ste=F,app=F)

summary(ar.sta)
```

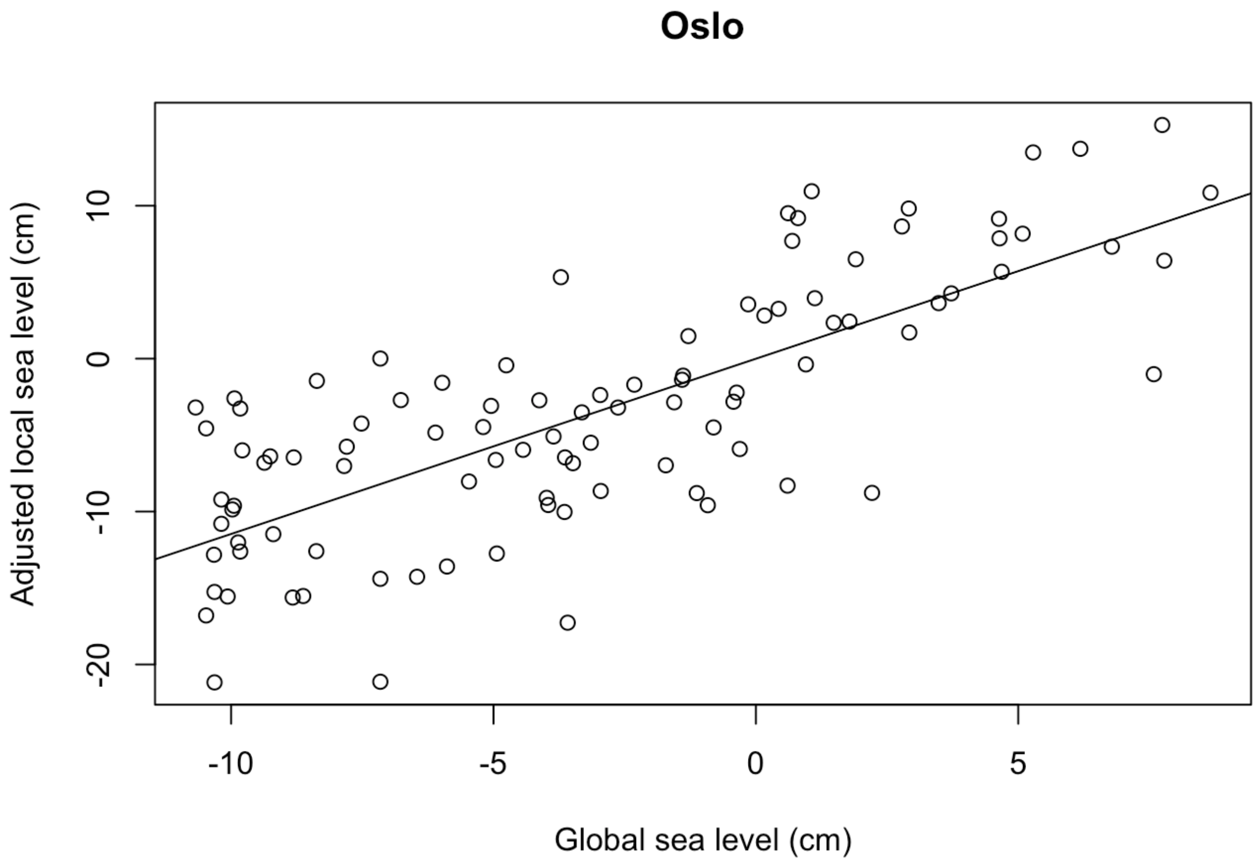
```
## Series: sta[sta.common.years]
## ARIMA(0,0,1) with zero mean
##
## Coefficients:
##          ma1  as.matrix(sealevel[sea.common.years])
##       0.2429                                 1.1465
## s.e.  0.0966                                 0.1038
##
## sigma^2 estimated as 26.01:  log likelihood=-298.76
## AIC=603.51   AICc=603.77   BIC=611.27
##
## Training set error measures:
##                     ME     RMSE     MAE      MPE    MAPE      MASE
## Training set 0.1476122 5.100249 4.11991 1508.879 1598.56 0.7879738
##                     ACF1
## Training set 0.001014127
```

```
order=ar.sta$arma[1]+ar.sta$arma[2]
ifelse(length(ar.sta$coef)-order==1,
{
a3= 0 #intercept 0
a4 = ar.sta$coef[order+1]
}
,
{
    a3=ar.sta$coef[order+1]
a4=ar.sta$coef[order+2] }
    )
```

```
## [1] 1.146478
```

Plot 5 Adjusted local series against global sea level with time series regression line

```
if(sum(is.element(plots,5))>0) {
    plot(sealevel[sea.common.years],sta[sta.common.years],xlab="Global sea level (
cm)",ylab="Adjusted local sea level (cm)",main=sta.name)
    abline(a3,a4)   }
```



Projections

For the models with temperature projections for all or specific RCPs, get paths to prepare for sea level prediction. Then calculate covariance structure and use it to produce simultaneous confidence bands for each scenario.

```r
models=switch(rcp,"all"=models.all,"2.6"=models2.6,"4.5"=models4.5,"6.0"=models6.0
,"8.5"=models8.5)

K = length(models)
modelpath1=matrix(0,K,101)
count=0
for (i in models) {
  count=count+1
  modelpath1[count,]=cmip5.data[proj,i,1]
 }

modelpath2=matrix(0,K,101)
count=0
for (i in models) {
  count=count+1
  modelpath2[count,]=cmip5.data[proj,i,2]
}

modelpath3=matrix(0,K,101)
count=0
for (i in models) {
  count=count+1
  modelpath3[count,]=cmip5.data[proj,i,3]
}

modelpath4=matrix(0,K,101)
count=0
for (i in models) {
  count=count+1
  modelpath4[count,]=cmip5.data[proj,i,4]
}


#Extract some parameters
n.o =  length(sta.common.years)
n.p=101
n = n.o+n.p

#Calculate covariance matrices for ARMA processes
cov2 = ARMAcov(armodel,n)
cov1 = ARMAcov(ar.sta,n)
Sigma2 = toeplitz(cov2)
Sigma1 = toeplitz(cov1)

#covariance for non-differentiated global mean sea level:
Sigma.i2 = apply(apply(Sigma2,1,cumsum),1,cumsum)
```

```
#Total covariance for station
Sigma = a4^2*Sigma.i2 + Sigma1

#add the variance caused by the uncertainty in the starting value:
Sigma = Sigma + (sd1999^2)*diag(dim(Sigma)[1])

#add the variance due to gia
times=(0:(n-1))-adj #must be in terms of anomalies from 1970-99 average
Sigma=Sigma+se.gia^2*times^2*diag(dim(Sigma)[1])

#Submatrices for observations and predictions:
Sigma.o = Sigma[1:n.o,1:n.o]
Sigma.p = Sigma[(n.o+1):n,(n.o+1):n]
Sigma.op = Sigma[1:n.o,(n.o+1):n]

#Calculate precision matrix
R.o = chol(Sigma.o)
c = mean(sealevel - cumsum(a1+a2*temp))
x.o = sta[1:n.o]#!!!correct
Q.o = chol2inv(R.o)
Sigma.pred = Sigma.p - t(Sigma.op) %*% Q.o %*% Sigma.op
Q.pred = Matrix(chol2inv(chol(Sigma.pred)))
sd = sqrt(diag(Sigma.pred))

#calculate the estimates for each climate model:
tmp = c+cumsum(a1+a2*temp[sta.common.years])
mu.o = a3+a4*tmp[sta.common.years]#!!!correct
mu.corr = t(Sigma.op) %*% (Q.o %*% (x.o - mu.o))
mu.pred1 = Matrix(0,K,n.p)
mu.pred2 = Matrix(0,K,n.p)
mu.pred3 = Matrix(0,K,n.p)
mu.pred4 = Matrix(0,K,n.p)

#gia backcorrection (i.e., uncorrected observed values)
sta=sta.un

mu.pred1 <- mu.pred2 <- mu.pred3 <- mu.pred4 <- Q <- list()
w = rep(1/K,K)
```

Here we calculate the actual predictions

```
for(k in 1:K){
  mu.pred1[[k]] = as.vector(a3 + a4*(cumsum(a1+a2*modelpath1[k,])) + s1999-gia*(re
g0+(0:100)))
  mu.pred2[[k]] = as.vector(a3 + a4*(cumsum(a1+a2*modelpath2[k,])) + s1999-gia*(re
g0+(0:100)))
  mu.pred3[[k]] = as.vector(a3 + a4*(cumsum(a1+a2*modelpath3[k,])) + s1999-gia*(re
g0+(0:100)))
  mu.pred4[[k]] = as.vector(a3 + a4*(cumsum(a1+a2*modelpath4[k,])) + s1999-gia*(re
g0+(0:100)))
  Q[[k]] = as(Q.pred,"sparseMatrix")
}
```

This calculates simultaneous and pointwise bands

```
res1 <- simconf.mixture(alpha=1-conf, mu = mu.pred1, Q = Q, w = w)
```

```
## in optimization:  0.0381966   0.0682   0.6918912
## in optimization:  0.0618034   0.0124   0.7878338
## in optimization:  0.0236068   0.1869   0.5085116
## in optimization:  0.0145898   0.3406   0.3129284
## in optimization:  0.009016994   0.4966   0.1627316
## in optimization:  0.005572809   0.6392   0.06801664
## in optimization:  0.003444185   0.7476   0.02322576
## in optimization:  0.002128624   0.8338   0.00438244
## in optimization:  0.001315562   0.8912   7.744e-05
## in optimization:  0.00109793   0.9049   2.401e-05
## in optimization:  0.001181689   0.8986   1.96e-06
## in optimization:  0.001222379   0.8962   1.444e-05
## in optimization:  0.001140999   0.9012   1.44e-06
## in optimization:  0.001140999   0.9012   1.44e-06
```

```
res2 <- simconf.mixture(alpha=1-conf, mu = mu.pred2, Q = Q, w = w)
```

```
## in optimization:  0.0381966   0.0632   0.7002342
## in optimization:  0.0618034   0.0124   0.7878338
## in optimization:  0.0236068   0.1792   0.5195526
## in optimization:  0.0145898   0.3366   0.3174196
## in optimization:  0.009016994   0.4972   0.1622478
## in optimization:  0.005572809   0.6421   0.06651241
## in optimization:  0.003444185   0.7546   0.02114116
## in optimization:  0.002128624   0.8403   0.00356409
## in optimization:  0.001315562   0.8952   2.304e-05
## in optimization:  0.001207385   0.9026   6.76e-06
## in optimization:  0.001166695   0.9061   3.721e-05
## in optimization:  0.001248075   0.8998   4e-08
## in optimization:  0.001248075   0.8998   4e-08
```

```
res3 <- simconf.mixture(alpha=1-conf, mu = mu.pred3, Q = Q, w = w)
```

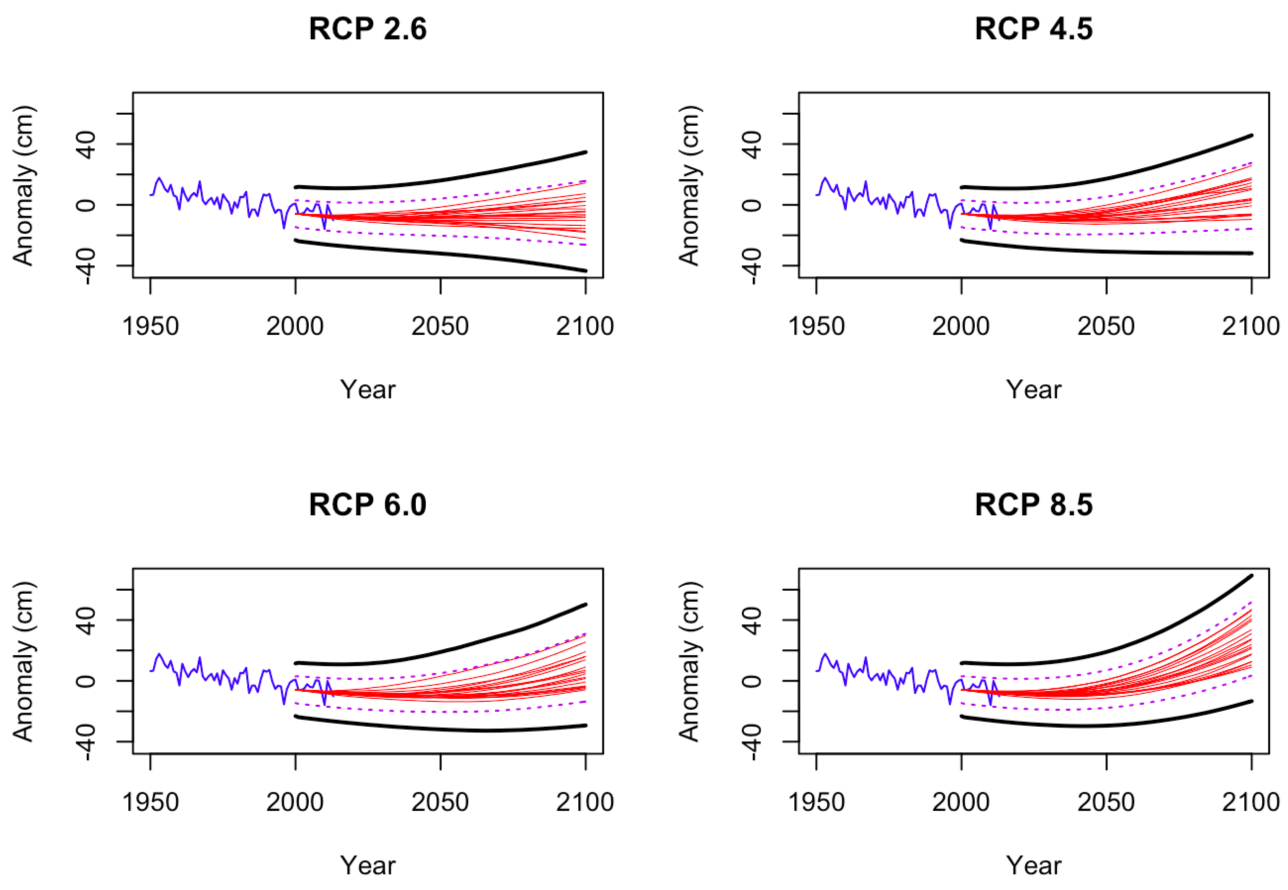
```
## in optimization:  0.0381966   0.0809   0.6709248
## in optimization:  0.0618034   0.0176   0.7786298
## in optimization:  0.0236068   0.2087   0.4778957
## in optimization:  0.0145898   0.37   0.2809
## in optimization:  0.009016994   0.5285   0.1380123
## in optimization:  0.005572809   0.655   0.060025
## in optimization:  0.003444185   0.758   0.020164
## in optimization:  0.002128624   0.8374   0.00391876
## in optimization:  0.001315562   0.889   0.000121
## in optimization:  0.00107458   0.9061   3.721e-05
## in optimization:  0.001152684   0.8999   1e-08
## in optimization:  0.001193374   0.8977   5.29e-06
## in optimization:  0.001152684   0.8999   1e-08
```

```
res4 <- simconf.mixture(alpha=1-conf, mu = mu.pred4, Q = Q, w = w)
```

```
## in optimization:  0.0381966   0.0684   0.6915586
## in optimization:  0.0618034   0.0144   0.7842874
## in optimization:  0.0236068   0.183   0.514089
## in optimization:  0.0145898   0.3377   0.3161813
## in optimization:  0.009016994   0.4918   0.1666272
## in optimization:  0.005572809   0.6343   0.07059649
## in optimization:  0.003444185   0.7413   0.02518569
## in optimization:  0.002128624   0.8247   0.00567009
## in optimization:  0.001315562   0.884   0.000256
## in optimization:  0.0008552213   0.9203   0.00041209
## in optimization:  0.001116242   0.8993   4.9e-07
## in optimization:  0.001075552   0.903   9e-06
## in optimization:  0.001156932   0.8957   1.849e-05
## in optimization:  0.001116242   0.8993   4.9e-07
```

Plot 6 Projected observations with 90% simultaneous confidence band

```
if(sum(is.element(plots,6))>0)
    {
if(rcp=="all") {
    par(mfrow=c(2,2))
    ylim=range(c(sta,res1$a,res1$b,res2$a,res2$b,res3$a,res3$b,res4$a,res4$b))
    }

    if(rcp=="2.6") {
        par(mfrow=c(1,1))
    ylim=range(c(sta,res1$a,res1$b))}

    if(rcp=="4.5") {
        par(mfrow=c(1,1))
    ylim=range(c(sta,res2$a,res2$b))}

    if(rcp=="6.0") {
        par(mfrow=c(1,1))
    ylim=range(c(sta,res3$a,res3$b))}

    if(rcp=="8.5") {
        par(mfrow=c(1,1))
    ylim=range(c(sta,res4$a,res4$b))}
begin=max(min(sta.common.years),which(is.element(sta.years,1950)))#index of1950 or
 earliest available date thereafter

if({rcp=="all"}||{rcp=="2.6"})
{
plot(c(sta.years[begin],2100),ylim,ylim=ylim,type='n',xlab="Year",
    ylab="Anomaly (cm)",main="RCP 2.6")
lines(sta.years[begin:max(sta.common.years)],sta[begin:max(sta.common.years)],col=
'blue')

for(ii in 1:K){
    lines(2000:2100,mu.pred1[[ii]],col=2,lwd=0.5)
}

lines(2000:2100,res1$b,lwd=2)
lines(2000:2100,res1$a,lwd=2)
lines(2000:2100,res1$a.marginal,col="purple",lty=3)
lines(2000:2100,res1$b.marginal,col="purple",lty=3)
}

if({rcp=="all"}||{rcp=="4.5"})
{
plot(c(sta.years[begin],2100),ylim,ylim=ylim,type='n',xlab="Year",
    ylab="Anomaly (cm)",main="RCP 4.5")
lines(sta.years[begin:max(sta.common.years)],sta[begin:max(sta.common.years)],col=
'blue')


for(ii in 1:K){
    lines(2000:2100,mu.pred2[[ii]],col=2,lwd=0.5)
}

lines(2000:2100,res2$b,lwd=2)
lines(2000:2100,res2$a,lwd=2)
lines(2000:2100,res2$a.marginal,col="purple",lty=3)
lines(2000:2100,res2$b.marginal,col="purple",lty=3)
}

if({rcp=="all"}||{rcp=="6.0"})
{

plot(c(sta.years[begin],2100),ylim,ylim=ylim,type='n',xlab="Year",
    ylab="Anomaly (cm)",main="RCP 6.0")
lines(sta.years[begin:max(sta.common.years)],sta[begin:max(sta.common.years)],col=
'blue')


for(ii in 1:K){
    lines(2000:2100,mu.pred3[[ii]],col=2,lwd=0.5)
}

lines(2000:2100,res3$b,lwd=2)
lines(2000:2100,res3$a,lwd=2)
lines(2000:2100,res3$a.marginal,col="purple",lty=3)
lines(2000:2100,res3$b.marginal,col="purple",lty=3)
}

if({rcp=="all"}||{rcp=="8.5"})
{
plot(c(sta.years[begin],2100),ylim,ylim=ylim,type='n',xlab="Year",
    ylab="Anomaly (cm)",main="RCP 8.5")
lines(sta.years[begin:max(sta.common.years)],sta[begin:max(sta.common.years)],col=
'blue')


for(ii in 1:K){
    lines(2000:2100,mu.pred4[[ii]],col=2,lwd=0.5)
}

lines(2000:2100,res4$b,lwd=2)
lines(2000:2100,res4$a,lwd=2)
lines(2000:2100,res4$a.marginal,col="purple",lty=3)
lines(2000:2100,res4$b.marginal,col="purple",lty=3)

}
}
```



One can also compute other quantities, such as

- probability to stay below a given level
- confidence bands for given years
- extreme value predictions