Brief Introduction to the Spatio-Temporal R-package

Initialisation

Load the package (along with other packages needed for the introduction):

```
> library(SpatioTemporal)
> library(plotrix)
> library(maps)
> library(fields)
```

Define plot functions that will be used to illustrate the data.

```
> ##create colour-scheme
> jet.colors <- colorRampPalette(c("#00007F", "blue", "#007FFF",
    "cyan", "#7FFF7F", "yellow", "#FF7F00", "red", "#7F0000"))
> ##Function that plots coloured points
> scatter <- function(x, y=NULL, value, colramp=jet.colors,
      legend=TRUE, clim=range(value,na.rm=TRUE), cres=256,
      add=FALSE, truncate=TRUE, ...){
    if( missing(y) || is.null(y) ){
      y <- x[,2]; x <- x[,1];
    }
    ##Compute colour scales.
    Ind <- round((cres-1)*(value-clim[1]) / (clim[2]-clim[1])+1)
    if(truncate){
      Ind[Ind<1] <- 1
      Ind[Ind>cres] <- cres
    }else{
      Ind[Ind<1 | Ind>cres] <- NA
    }
    ##match length to the x and y data.
      Ind <- c(Ind,rep(NA,length(x)-length(Ind)))
    ##Do the plots
    if(add){
      points(x, y, col=jet.colors(cres)[Ind], ...)
    }else{
      plot(x, y, col=jet.colors(cres)[Ind], ...)
    }
    if(legend){
      image.plot(legend.only=TRUE, zlim=clim,
                 col=jet.colors(cres))
    }
  }
```

Loading the data

As an example we'll study NO_x data from Los Angeles, first we load the raw data

```
> data(mesa.data.raw)
> names(mesa.data.raw)
```

```
[1] "X"             "obs"           "lax.conc.1500"
```

The data consists of observations (log of NO_x concentrations), in a time-by-location matrix with missing NA

```
> head(mesa.data.raw$obs)
```

```
           60370002 60370016 60370030 60370031 60370113
1999-01-13 4.577684 4.131632       NA       NA 4.727882
1999-01-27 3.889091 3.543566       NA       NA 4.139332
1999-02-10 4.013020 3.632424       NA       NA 4.054051
1999-02-24 4.080691 3.842586       NA       NA 4.392799
1999-03-10 3.728085 3.396944       NA       NA 3.960577
1999-03-24 3.751913 3.626161       NA       NA 3.958741
           60371002 60371103 60371201 60371301 60371601
1999-01-13 5.352608 5.281452 4.984585 5.463134 5.316398
1999-01-27 4.876832 4.846044 4.100073 5.213077 5.010987
1999-02-10 4.717611 4.665429 4.056365 5.037477 4.770632
1999-02-24 4.877139 4.830275 4.382803 5.127157 4.960104
1999-03-10 4.252480 4.163820 3.808937 4.656825 4.205851
1999-03-24 4.180627 4.240120 3.794791 4.583794 4.383694
           60371602 60371701 60372005 60374002 60375001
1999-01-13       NA 5.081886 4.900640 4.995868 5.165070
1999-01-27       NA 4.674858 4.381561 4.785056 4.784252
1999-02-10       NA 4.715861 4.247208 4.493267 4.685089
1999-02-24       NA 4.905827 4.450186 4.440054 4.676942
1999-03-10       NA 4.403685 3.792204 4.035339 4.030772
1999-03-24       NA 4.472207 3.836844 3.995005 4.200838
           60375005 60590001 60590007 60591003 60595001
1999-01-13       NA 4.847385       NA 4.603461 4.834629
1999-01-27       NA 4.517424       NA 4.414679 4.576023
1999-02-10       NA 4.217816       NA 4.104592 4.337169
1999-02-24       NA 4.565771       NA 4.288501 4.573462
1999-03-10       NA 3.816688       NA 3.374445 3.936019
1999-03-24       NA 3.795629       NA 3.412111 3.914319
           L001 L002 LC001 LC002 LC003
1999-01-13   NA   NA    NA    NA    NA
1999-01-27   NA   NA    NA    NA    NA
1999-02-10   NA   NA    NA    NA    NA
1999-02-24   NA   NA    NA    NA    NA
1999-03-10   NA   NA    NA    NA    NA
1999-03-24   NA   NA    NA    NA    NA
```

as well as location and covariate information for the 25 sites

```
> head(mesa.data.raw$X)
```

```
        ID         x        y     long     lat type
1 60370002 -10861.67 3793.589 -117.923 34.1365  AQS
2 60370016 -10854.95 3794.456 -117.850 34.1443  AQS
3 60370030 -10888.66 3782.332 -118.216 34.0352  AQS
4 60370031 -10891.42 3754.649 -118.246 33.7861  AQS
5 60370113 -10910.76 3784.099 -118.456 34.0511  AQS
6 60371002 -10897.96 3797.979 -118.317 34.1760  AQS
  log10.m.to.a1 log10.m.to.a2 log10.m.to.a3 log10.m.to.road
1      2.861509      4.100755      2.494956        2.494956
2      3.461672      3.801059      2.471498        2.471498
3      2.561133      3.695772      1.830197        1.830197
4      3.111413      2.737527      2.451927        2.451927
5      2.762193      3.687412      2.382281        2.382281
6      2.760931      4.035977      1.808260        1.808260
  km.to.coast s2000.pop.div.10000
1   15.000000            1.733283
2   15.000000            1.645386
3   15.000000            6.192630
4    1.023311            2.088930
5    6.011075            7.143731
6   15.000000            4.766780
```

and a spatio-temporal covariate (not used here).

Creating an `STdata`-object

Our first step is now to collect the available data into a suitable data structure.

```
> ##extract observations and covariates
> obs <- mesa.data.raw$obs
> covars <- mesa.data.raw$X
> ##create STdata object
> mesa.data <- createSTdata(obs, covars)
```

The resulting structure contains information about the monitoring sites (location and covariates) as well as the observations.

```
> print(mesa.data)

STdata-object with:
        No. locations: 25 (observed: 25)
        No. time points: 280 (observed: 280)
        No. obs: 4577

No trend specified

12 covariate(s):
```

```
 [1] "ID"                  "x"
 [3] "y"                   "long"
 [5] "lat"                 "type"
 [7] "log10.m.to.a1"       "log10.m.to.a2"
 [9] "log10.m.to.a3"       "log10.m.to.road"
[11] "km.to.coast"         "s2000.pop.div.10000"

No spatio-temporal covariates.

All sites:
  AQS FIXED
   20     5
Observed:
  AQS FIXED
   20     5

For AQS:
  Number of obs: 4178
  Dates: 1999-01-13 to 2009-09-23
For FIXED:
  Number of obs: 399
  Dates: 2005-12-07 to 2009-07-01
```

We can, for example, study the times and locations at which observations where obtained (Fig.~1),

```
> plot(mesa.data, "loc")
```

as well as the spatial locations of the observations (Fig.~2).

```
> plot(mesa.data$covars$long, mesa.data$covars$lat,
       pch=c(24,25)[mesa.data$covars$type],
       bg=c("red","blue")[mesa.data$covars$type],
       xlab="Longitude", ylab="Latitude")
> ##Add the map of LA
> map("county", "california", col="#FFFF0055", fill=TRUE,
      add=TRUE)
> ##Add a legend
> legend("bottomleft", c("AQS","FIXED"), pch=c(24,25), bty="n",
         pt.bg=c("red","blue"))
```

Smooth temporal trends

The first step in analysing the data is to determine how many smooth trends are needed to capture the seasonal variability (Fig.~3).

```
> ##extract a data matrix
> D <- createDataMatrix(mesa.data)
```

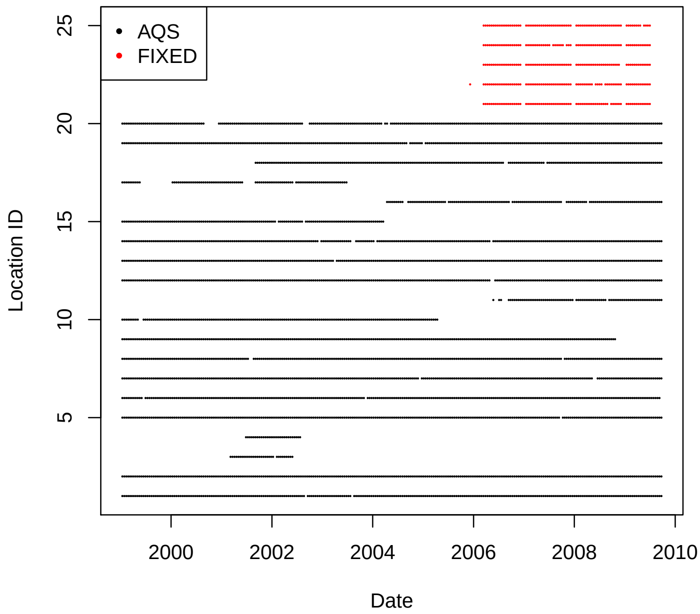

Figure 1: Space-time location of all our observations. Note that the FIXED, i.e. MESA specific monitors, only sampled during the second half of the period.

```
> ##Run leave one out cross-validation to find smooth trends
> SVD.cv <- SVDsmoothCV(D,1:5)
> ##Study the results
> print(SVD.cv)

Result of SVDsmoothCV, summary of cross-validation:
               RMSE        R2       BIC
n.basis.1 0.2641928 0.8666995 -11763.23
n.basis.2 0.2309584 0.8981274 -12783.19
n.basis.3 0.2286814 0.9001262 -12663.16
n.basis.4 0.2233561 0.9047236 -12668.14
n.basis.5 0.2233990 0.9046870 -12455.66

Individual BIC:s for each column:
        n.basis.1 n.basis.2 n.basis.3 n.basis.4 n.basis.5
Min.      -913.20    -968.3   -965.10   -963.70   -955.00
1st Qu.   -806.20    -820.5   -820.80   -827.40   -825.90
Median    -553.50    -571.3   -564.80   -583.90   -565.40
Mean      -504.10    -538.3   -537.20   -541.10   -535.90
3rd Qu.   -248.40    -271.9   -263.00   -260.10   -255.20
Max.       -83.82     -86.1    -82.78    -84.67    -82.24
```

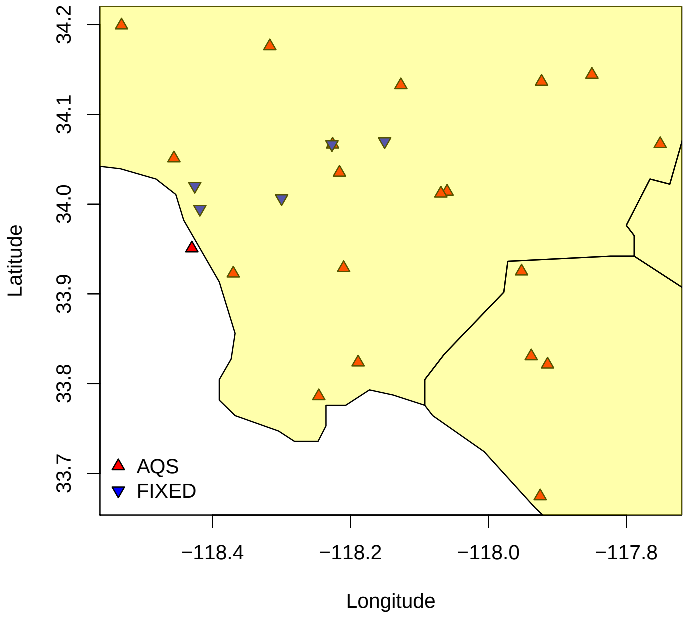

Figure 2: Location of monitors in the Los Angeles area.

```
> ##plot the cv-results
> plot(SVD.cv)
```

However just looking at overall statistics might be misleading. We can also do scatter plots of BIC for different number of trends at all the sites (Fig.~4).

```
> plot(SVD.cv,pairs=TRUE)
```

We now add two smooth temporal trends to the data structure.

```
> mesa.data <- updateSTdataTrend(mesa.data, n.basis = 2)
> print(mesa.data)

STdata-object with:
        No. locations: 25 (observed: 25)
        No. time points: 280 (observed: 280)
        No. obs: 4577

Trend with 2 basis function(s):
[1] "V1" "V2"
with dates:
        1999-01-13 to 2009-09-23
```

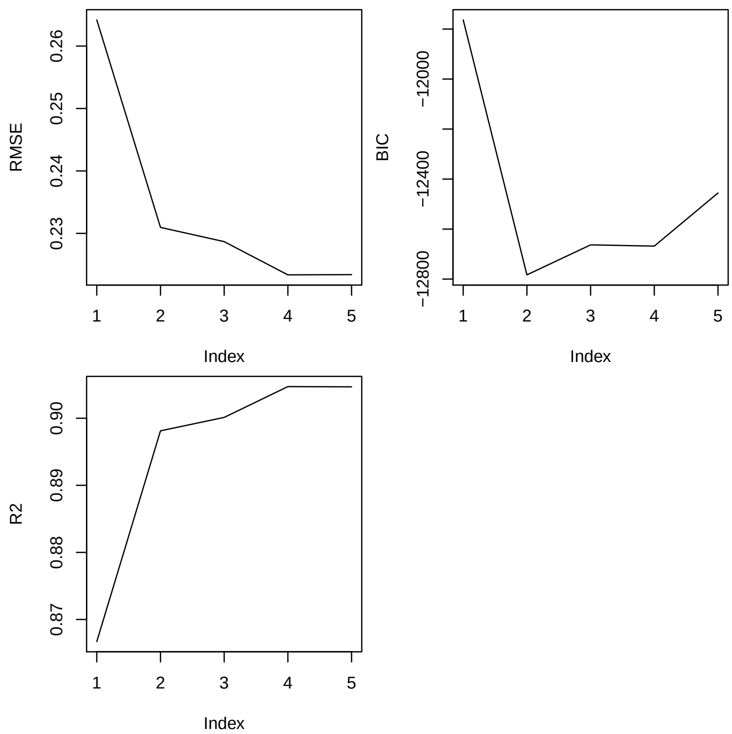

Figure 3: Cross-validation results for different numbers of smooth temporal trends.

```
12 covariate(s):
 [1] "ID"                   "x"
 [3] "y"                    "long"
 [5] "lat"                  "type"
 [7] "log10.m.to.a1"        "log10.m.to.a2"
 [9] "log10.m.to.a3"        "log10.m.to.road"
[11] "km.to.coast"          "s2000.pop.div.10000"

No spatio-temporal covariates.

All sites:
  AQS FIXED
   20     5
Observed:
  AQS FIXED
   20     5

For AQS:
  Number of obs: 4178
  Dates: 1999-01-13 to 2009-09-23
For FIXED:
```

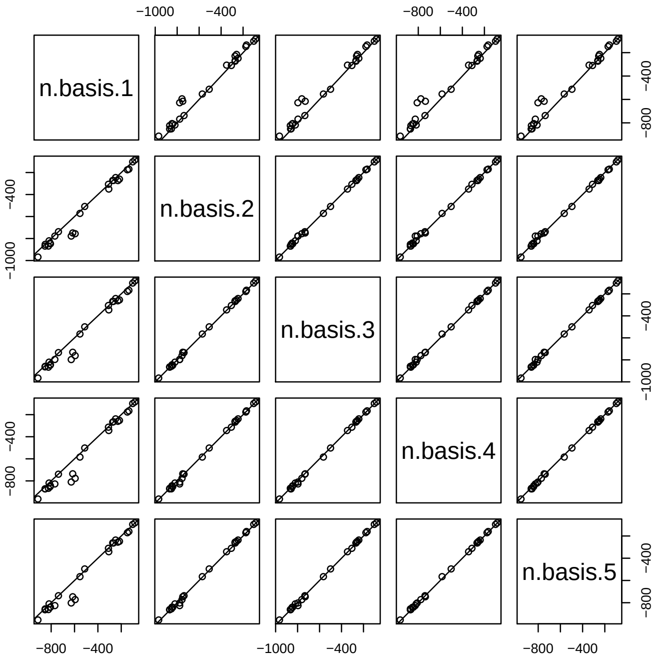

Figure 4: BIC at all sites for different numbers of temporal trends. Note that as we increase the number of trends all sites don't behave equally. Some sites require many trends and some few.

```
Number of obs: 399
Dates: 2005-12-07 to 2009-07-01
```

Given smooth trends we fit the observations to the trends at each site, and study the residuals (Fig.~5).

```
> ##plot observations at some of the locations with the
> ##fitted smooth trends
> par(mfcol=c(4,1),mar=c(2.5,2.5,2,.5))
> plot(mesa.data, "obs", ID=5)
> plot(mesa.data, "res", ID=5)
> plot(mesa.data, "obs", ID=18)
> plot(mesa.data, "res", ID=18)
```

Since we want the temporal trends to capture the temporal variability we also study the auto correlation function of the residuals to determine how much temporal dependence remains after fitting the temporal trends (Fig.~6).

```
> par(mfcol=c(2,2),mar=c(2.5,2.5,3,.5))
> plot(mesa.data, "acf", ID=1)
```

```
> plot(mesa.data, "acf", ID=5)
> plot(mesa.data, "acf", ID=13)
> plot(mesa.data, "acf", ID=18)
```

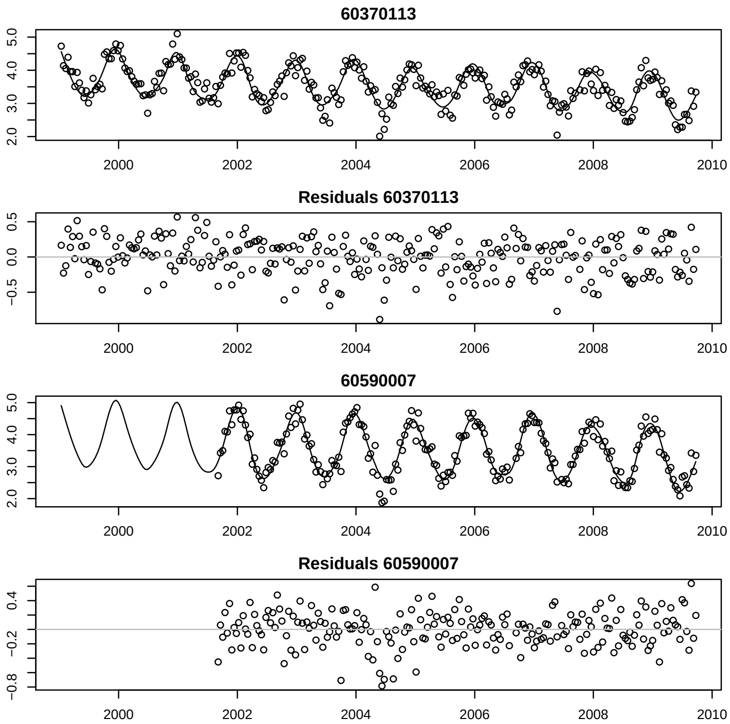

Figure 5: Smooth temporal trends and residuals for two locations.

β–fields

Given smooth temporal trends we fit each of the times series of observations to the smooth trends and extract the regression coefficients

```
> ##extract data
> D <- createDataMatrix(mesa.data)
> ##create matrix that holds estimated beta:s
> beta <- matrix(NA, dim(D)[2], dim(mesa.data$trend)[2])
> beta.std <- beta
> ##get the trends
> F <- mesa.data$trend
> ##this includes a data column, that we drop
> F$date <- NULL
> ##linear regression of observations on the trends
> for(i in 1:dim(D)[2]){
    tmp <- lm(D[,i] ~ as.matrix(F))
    beta[i,] <- coef(tmp)
```

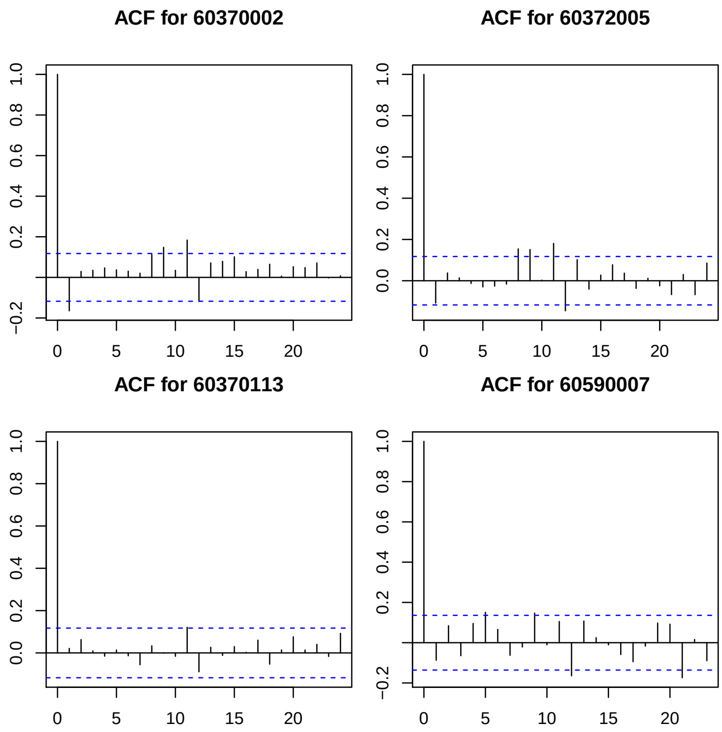

Figure 6: Auto-correlation functions for four locations.

```
    beta.std[i,] <- sqrt(diag(vcov(tmp)))
  }
> ##Add names to the estimated betas
> colnames(beta) <- c("beta.0","beta.1","beta.2")
> rownames(beta) <- colnames(D)
```

In the full spatio-temporal model these β–fields are modelled using geographic covariates. Selection of covariates is done by comparing these fields to the available covariates. However this is outside the scope of this introduction. For now we look at the spatial distribution of the regression coefficients (Fig.˜7) and keep the values so that we can (eventually) compare them to the results from the full model.

```
> par(mfcol=c(2,2))
> for(i in 1:3){
    scatter(mesa.data$covars$long, mesa.data$covars$lat,
            beta[,i], pch=19)
    map("county","california",add=TRUE)
  }
```

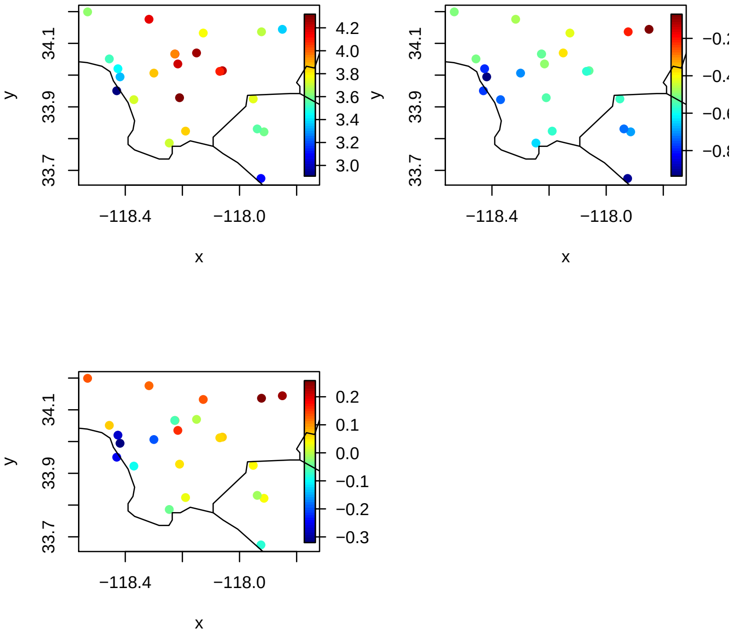

Figure 7: Spatial distribution of the β:s.

Estimating the model

Given the available covariates

```
> names(mesa.data$covars)

 [1] "ID"                    "x"
 [3] "y"                     "long"
 [5] "lat"                   "type"
 [7] "log10.m.to.a1"         "log10.m.to.a2"
 [9] "log10.m.to.a3"         "log10.m.to.road"
[11] "km.to.coast"           "s2000.pop.div.10000"
```

we create a model with three covariates for the temporal intercept, one covariate for the two temporal trends, and no spatio-temporal covariates; exponential covariances for the β and ν-fields; we also specify which covariates to use as locations for our observations.

```
> ##define land-use covariates
> LUR <-  list(c("log10.m.to.a1", "s2000.pop.div.10000",
                 "km.to.coast"), "km.to.coast", "km.to.coast")
> ##and covariance model
> cov.beta <- list(covf="exp", nugget=FALSE)
```

```
> cov.nu <- list(covf="exp", nugget=TRUE, random.effect=FALSE)
> ##which locations to use
> locations <- list(coords=c("x","y"), long.lat=c("long","lat"),
                    others="type")
> ##create object
> mesa.model <- createSTmodel(mesa.data, LUR=LUR, ST=NULL,
                              cov.beta=cov.beta, cov.nu=cov.nu,
                              locations=locations)
> ##inspect the resulting model
> print(mesa.model)

STmodel-object with:
        No. locations: 25 (observed: 25)
        No. time points: 280 (observed: 280)
        No. obs: 4577

Trend with 2 basis function(s):
[1] "V1" "V2"
with dates:
        1999-01-13 to 2009-09-23

Models for the beta-fields are:
$const
~log10.m.to.a1 + s2000.pop.div.10000 + km.to.coast

$V1
~km.to.coast

$V2
~km.to.coast

No spatio-temporal covariates.

Covariance model for the beta-field(s):
        Covariance type(s): exp, exp, exp
        Nugget: No, No, No
Covariance model for the nu-field(s):
        Covariance type: exp
        Nugget: ~1
        Random effect: No
All sites:
  AQS FIXED
   20     5
Observed:
  AQS FIXED
   20     5

For AQS:
  Number of obs: 4178
  Dates: 1999-01-13 to 2009-09-23
```

```
For FIXED:
  Number of obs: 399
  Dates: 2005-12-07 to 2009-07-01
```

Given the model we setup initial values for the optimisation. Here we're using two different starting points

```
> ##Some important dimensions of the model
> dim <- loglikeSTdim(mesa.model)
> x.init<-cbind(rep(2,dim$nparam.cov),
                c(rep(c(1,-3),dim$m+1),-3))
```

We are now ready to estimate the model.

DO NOT RUN!!!

However this takes a rather long time

```
> ##estimate parameters
> est.mesa.model <- estimate(mesa.model, x.init,
                             hessian.all=TRUE)
```

Run this instead

Instead we load the precomputed results

```
> data(est.mesa.model)
```

Evaluating the results

Having estimated the model we studying the results, taking special note of the status message that indicates if the optimisation has converged.

```
> print(est.mesa.model)

Optimisation for STmodel with 2 starting points.
  Results: 2 converged, 0 not converged, 0 failed.
  Best result for starting point 1, optimisation has converged

No fixed parameters.

Estimated parameters for all starting point(s):
                                   [,1]        [,2]
alpha.const.(Intercept)         3.76839891  3.76876752
alpha.const.log10.m.to.a1      -0.21091617 -0.21104622
alpha.const.s2000.pop.div.10000  0.04060424  0.04058733
alpha.const.km.to.coast         0.03780535  0.03782602
```

```
alpha.V1.(Intercept)           -0.74465855 -0.74298680
alpha.V1.km.to.coast            0.01752644  0.01739711
alpha.V2.(Intercept)           -0.14007845 -0.14049831
alpha.V2.km.to.coast            0.01609611  0.01611804
log.range.const.exp             2.27189693  2.26718139
log.sill.const.exp             -2.74917305 -2.75153241
log.range.V1.exp                2.89644168  2.92490516
log.sill.V1.exp                -3.53320957 -3.51478747
log.range.V2.exp                1.54547660  1.53480791
log.sill.V2.exp                -4.62603138 -4.62108099
nu.log.range.exp                4.41060631  4.41052883
nu.log.sill.exp                -3.22902927 -3.22911940
nu.log.nugget.(Intercept).exp  -4.30036905 -4.30029808

Function value(s):
[1] 5732.202 5732.202
```

We then plot the estimated parameters (Fig.~8), along with approximate confidence intervals from the observed information matrix.

```
> par <- est.mesa.model$res.best$par.all
> par(mfrow=c(1,1),mar=c(13,2.5,.5,.5))
> plotCI(par$par, uiw=1.96*par$sd, ylab="", xlab="", xaxt="n")
> axis(1, 1:dim(par)[1], rownames(par), las=2)
```

Having estimate the model parameters we can now compute the conditional expectations of the observed locations and latent β-fields

```
> EX <- predict(mesa.model, est.mesa.model$res.best$par, pred.var = TRUE)
```

The predictions can be used to extend the shorter time-series to predictions covering the entire period. To illustrate we plot predictions and observations for 4 different locations (Fig.~9).

```
> par(mfrow=c(4,1),mar=c(2.5,2.5,2,.5))
> plot(EX, ID=1, STmodel=mesa.model, pred.var=TRUE)
> plot(EX, ID=10, STmodel=mesa.model, pred.var=TRUE)
> plot(EX, ID=17, STmodel=mesa.model, pred.var=TRUE)
```

Alternatively we can also study the predictions due to different parts of the model

```
> plot(EX, ID=17, STmodel=mesa.model, pred.var=TRUE, lwd=2)
> plot(EX, ID=17, pred.type="EX.mu", col="green", add=TRUE, lwd=2)
> plot(EX, ID=17, pred.type="EX.mu.beta", col="blue", add=TRUE, lwd=2)
```

e.g. just the linear regression (mean value part) for the β-fields, the universall kriging for the β-fields, or the full model including the ν-fields.

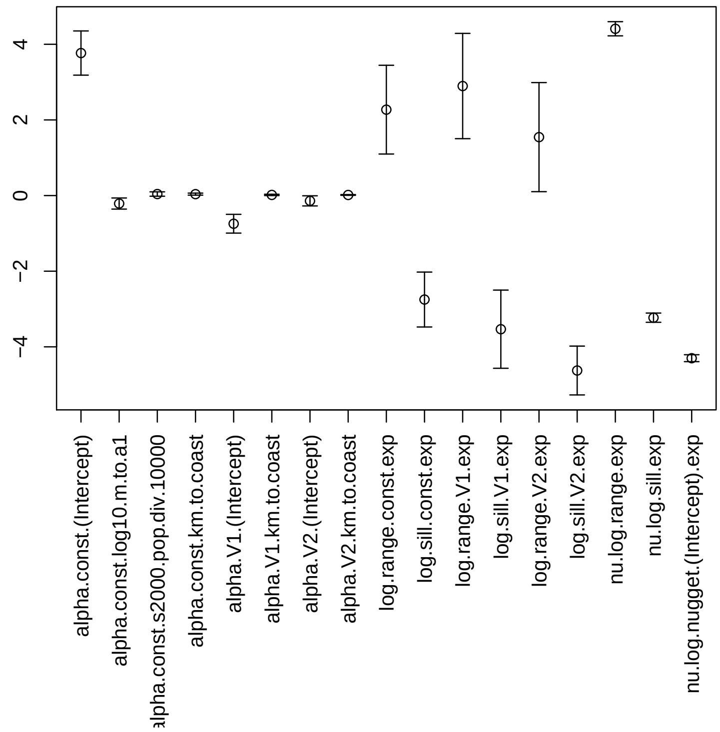

Figure 8: Estimated parameters and their 95% confidence intervals from the first starting point. Note which parameters have the smallest confidence intervals, any idea why?

We also compare the β-fields obtained from the full model with those previously computed by individually fitting each times series of observations to the smooth trends (Fig.~10).

```
> par(mfcol=c(2,2),mar=c(4.5,4.5,2,.5))
> for(i in 1:3){
    plotCI(x=beta[,i], y=EX$beta$EX[,i], pch=NA,
           uiw=1.96*sqrt( diag(EX$beta$VX[,i]) ),
           main=colnames(EX$beta$EX)[i],
           xlab="Empirical estimate",
           ylab="Spatio-Temporal Model")
    plotCI(x=beta[,i], y=EX$beta$EX[,i], pch=NA,
           uiw=1.96*beta.std[,i], err="x", add=TRUE)
    points(beta[,i], EX$beta$EX[,i], pch=19, cex=1, col="red")
    abline(0,1)
  }
```

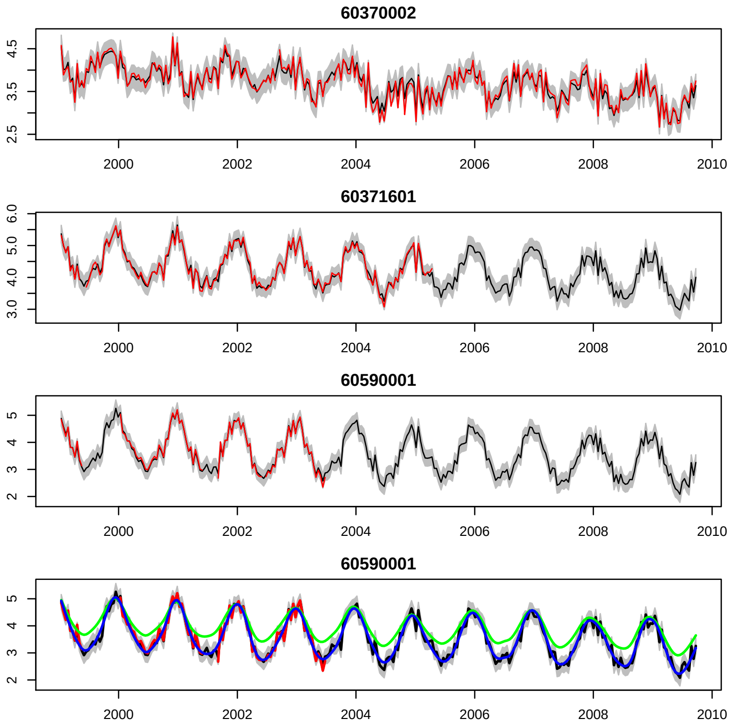

Figure 9: Predictions at 3 different locations, and the contributions from each part of the model.

Cross-validation

A cross-validation (CV) study is a simple but good way of evaluating model performance. First we define 10 CV groups, and study the number of observations in each group

```
> Ind.cv <- createCV(mesa.model, groups=10, min.dist=.1)
> table(Ind.cv)

Ind.cv
  1   2   3   4   5   6   7   8   9  10
438 389 811 556 546 165 228 487 160 797
```

And illustrate the location of sites that belong to the same CV groups (Fig.~11)

```
> I.col <- sapply(split(mesa.model$obs$ID, Ind.cv), unique)
> I.col <- apply(sapply(I.col, function(x) mesa.model$locations$ID
                  %in% x), 1, function(x) if(sum(x)==1) which(x) else 0)
> plot(mesa.model$locations$long, mesa.model$locations$lat,
       pch=23+floor(I.col/max(I.col)+.5), bg=I.col)
> map("county","california",add=TRUE)
```

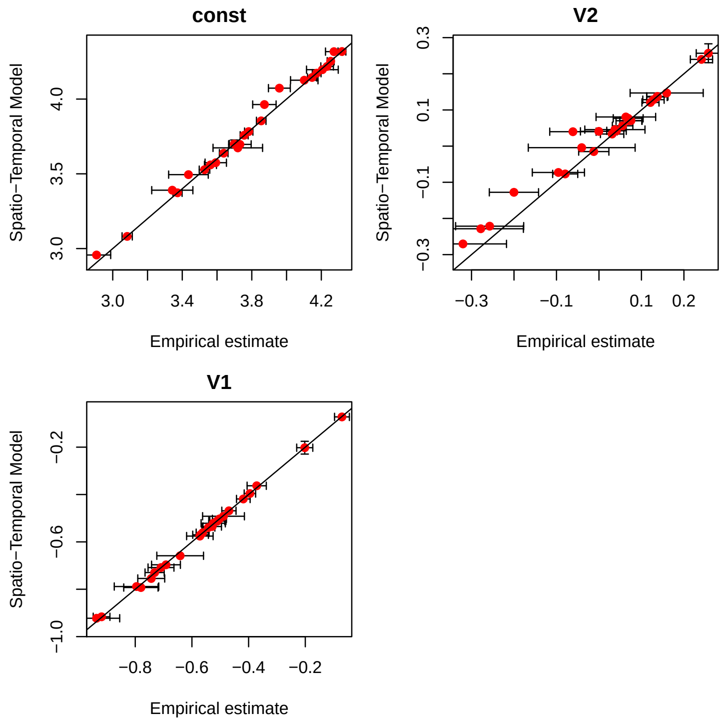

Figure 10: Comparisson of the two different estimates for the β-fields.

The CV functions, `estimateCV` and `predictCV`, will leave out observations marked by the current CV-groups number in the vector `Ind.cv`. For the first CV-groupd only observations such that `Ind.cv!=1` are used for parameter estimation, predictions are then done for the observations with `Ind.cv==1` given observations in `Ind.cv!=1` and the estimated parameters.

DO NOT RUN!!!

Estimated parameters and predictions for the 10-fold CV are obtained using:

```
> ##estimate different parameters for each CV-group
> est.cv.mesa <- estimateCV(mesa.model, x.init, Ind.cv)
> ##compute predictions at the different sites,
> ##given the estimated parameters
> pred.cv.mesa <- predictCV(mesa.model, est.cv.mesa$par.cov,
                            est.cv.mesa$Ind.cv)
```

Run this instead

However this takes a rather long time, so we load the precomputed results instead

```
> data(CV.mesa.model)
```

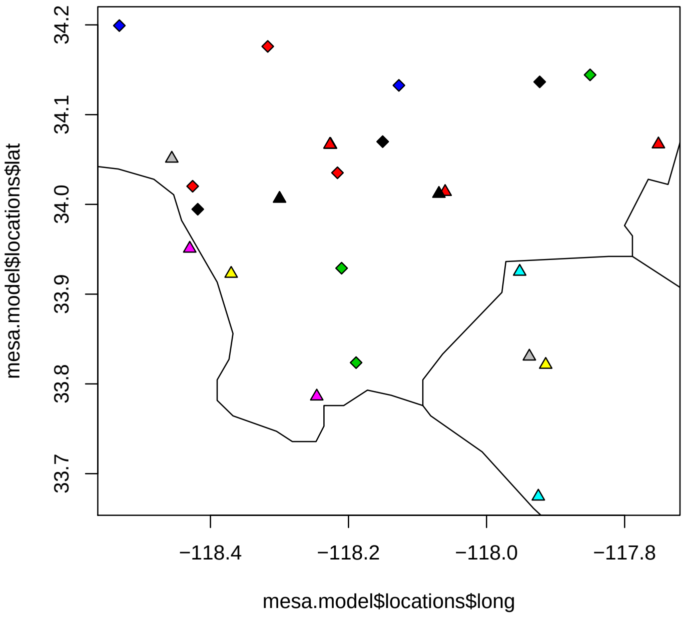

Figure 11: Locations of the CV-groups, sites that share the same symbol and colour belong to the same group.

Evaluating the results

First we examine the parmeter estimates,

```
> print(est.cv.mesa)

Cross-validation parameter estimation for STmodel
  with 10 CV-groups and 2 starting points.
  Results: 10 converged, 0 not converged.

No fixed parameters.

Estimated function values and convergence info:
      value convergence conv  eigen.min eigen.all.min
1  5170.173        TRUE TRUE 0.18601707            NA
2  5165.890        TRUE TRUE 1.54279413            NA
3  4681.795        TRUE TRUE 0.08526751            NA
4  4930.647        TRUE TRUE 1.07685472            NA
5  4986.502        TRUE TRUE 1.59698149            NA
6  5731.382        TRUE TRUE 0.18645515            NA
7  5409.620        TRUE TRUE 0.17342805            NA
8  5162.690        TRUE TRUE 0.92013448            NA
```

```
9  5487.728        TRUE TRUE 0.24066962            NA
10 4476.118        TRUE TRUE 1.09891407            NA
```

noting that the estimates for all 10 CV-groups have converged. We then compare the parameter estimates with those obtained when using all the data to fit the model (Fig.~12).

```
> par(mfrow=c(1,1), mar=c(13,2.5,.5,.5), las=2)
> boxplot(est.cv.mesa, plot.type="all")
> ##we've previously extracted the estimated parameters
> points(par$par, col=2, pch=19)
```

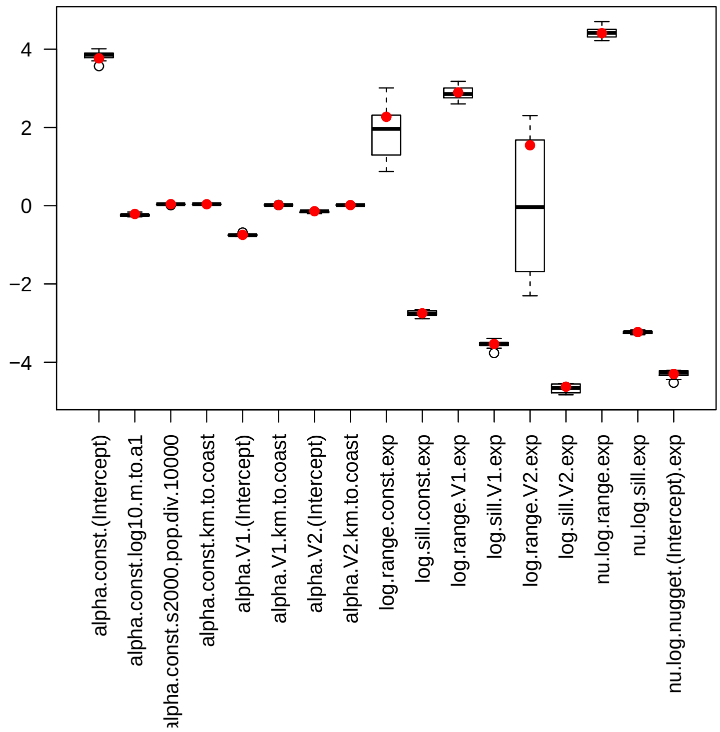

Figure 12: Parameters estimated from CV, compared with parameter estimates based on the full data-set.

To assess the models predictive ability we plot a couple of predicted timeseries (with 95% confidence intervals), and the left out observations (Fig.~**??**).

```
> ##look at the predictions at 4 sites
> par(mfcol=c(4,1),mar=c(2.5,2.5,2,.5))
> plotCV(pred.cv, 1, mesa.data.model)
> plotCV(pred.cv, 5, mesa.data.model)
> plotCV(pred.cv, 13, mesa.data.model)
> plotCV(pred.cv, 18, mesa.data.model)
```

We can also compute the root mean squared error, R^2, and coverage of 95% confidence intervals for the predictions.

```
> summary(pred.cv.mesa)

Cross-validation predictions for STmodel with 10 CV-groups.
  Predictions for 4577 observations.

RMSE:
        EX.mu EX.mu.beta        EX
obs 0.4302009    0.37162 0.3325597

R2:
        EX.mu EX.mu.beta        EX
obs 0.6465463  0.7362526 0.7887829

Coverage of 95% prediction intervalls:
           EX
obs 0.9217828
```

Another option is to do a scatter plot of the left out data against the predicted (points colour-coded by site)

```
> par(mfcol=c(1,1),mar=c(4.5,4.5,2,.5))
> plot(pred.cv$pred.obs[,"obs"], pred.cv$pred.obs[,"pred"],
       xlab="observations",ylab="predictions",pch=19,cex=.1,
       col=mesa.data.model$obs$idx)
> abline(0,1)
```

The end!