

# Spatial factor analysis

James T. Thorson

```
library(tinyVAST)
library(fmesher)
set.seed(101)
```

tinyVAST is an R package for fitting vector autoregressive spatio-temporal (VAST) models. We here explore the capacity to specify a spatial factor analysis, where the spatial pattern for multiple variables is described via their estimated association with a small number of spatial latent variables.

## Spatial factor analysis

We first explore the ability to specify two latent variables for five manifest variables. To start we simulate two spatial latent variables, project via a simulated loadings matrix, and then simulate a Tweedie response for each manifest variable:

```
# Simulate settings
theta_xy = 0.4
n_x = n_y = 10
n_c = 5
rho = 0.8
resid_sd = 0.5

# Simulate GMRFs
R_s = exp(-theta_xy * abs(outer(1:n_x, 1:n_y, FUN="-"))) )
R_ss = kronecker(X=R_s, Y=R_s)
delta_fs = mvtnorm::rmvnorm(n_c, sigma=R_ss )

#
L_cf = matrix( rnorm(n_c^2), nrow=n_c )
L_cf[,3:5] = 0
L_cf = L_cf + resid_sd * diag(n_c)

#
d_cs = L_cf %*% delta_fs
```

Where we can inspect the simulated loadings matrix

```
dimnames(L_cf) = list( paste0("Var ", 1:nrow(L_cf)),
                       paste0("Factor ", 1:ncol(L_cf)) )
knitr::kable( L_cf,
               digits=2, caption="True loadings")
```

Table 1: True loadings

|       | Factor 1 | Factor 2 | Factor 3 | Factor 4 | Factor 5 |
|-------|----------|----------|----------|----------|----------|
| Var 1 | -0.77    | 0.26     | 0.0      | 0.0      | 0.0      |
| Var 2 | -0.66    | 1.03     | 0.0      | 0.0      | 0.0      |
| Var 3 | -0.30    | -0.54    | 0.5      | 0.0      | 0.0      |
| Var 4 | -0.57    | 0.34     | 0.0      | 0.5      | 0.0      |
| Var 5 | -0.03    | -0.24    | 0.0      | 0.0      | 0.5      |

We then specify the model as expected by *tinyVAST*:

```
# Shape into longform data-frame and add error
Data = data.frame( expand.grid(species=1:n_c, x=1:n_x, y=1:n_y), "var"="logn", z=exp(as.vector(d_cs)) )
Data$n = tweedie::rtweedie( n=nrow(Data), mu=Data$z, phi=0.5, power=1.5 )
mean(Data$n==0)
#> [1] 0.03

# make mesh
mesh = fm_mesh_2d( Data[,c('x','y')] )

#
sem = "
  f1 -> 1, 11
  f1 -> 2, 12
  f1 -> 3, 13
  f1 -> 4, 14
  f1 -> 5, 15
  f2 -> 2, 16
  f2 -> 3, 17
  f2 -> 4, 18
  f2 -> 5, 19
  f1 <-> f1, NA, 1
  f2 <-> f2, NA, 1
  1 <-> 1, NA, 0
  2 <-> 2, NA, 0
  3 <-> 3, NA, 0
  4 <-> 4, NA, 0
  5 <-> 5, NA, 0
"

# fit model
out = fit( sem = sem,
  data = Data,
  formula = n ~ 0 + factor(species),
  spatial_graph = mesh,
  family = list( "obs"=tweedie() ),
  variables = c( "f1", "f2", 1:n_c ),
  data_colnames = list(spatial = c("x","y"), variable = "species", time = "time", distribution = "logn"),
  control = tinyVASTcontrol(quiet=TRUE, trace=0, gmrf="proj") )

out
#> $call
#> fit(data = Data, formula = n ~ 0 + factor(species), sem = sem,
#>     family = list(obs = tweedie()), data_colnames = list(spatial = c("x",
```

```

#>      "y"), variable = "species", time = "time", distribution = "dist"),
#>      variables = c("f1", "f2", 1:n_c), spatial_graph = mesh, control = tinyVASTcontrol(quiet = TRUE,
#>      trace = 0, gmrf = "proj"))
#>
#> $opt
#> $opt$par
#>      alpha_j      alpha_j      alpha_j      alpha_j      alpha_j      theta_z      theta_z      theta_z
#> 0.07570783 -0.02014888 0.22318277 0.14728087 -0.26514652 0.68016356 0.68285927 0.31701846 0.5
#>      theta_z      theta_z      theta_z      log_sigma      log_sigma      log_kappa
#> -0.31998633 0.23601680 -0.21613586 -0.52205331 0.21851154 -0.26761196
#>
#> $opt$objective
#> [1] 631.3721
#>
#> $opt$convergence
#> [1] 0
#>
#> $opt$iterations
#> [1] 71
#>
#> $opt$evaluations
#> function gradient
#>      84      71
#>
#> $opt$message
#> [1] "relative convergence (4)"
#>
#>
#> $sdrep
#> sdreport(.) result
#>      Estimate Std. Error
#> alpha_j      0.07570783 0.31850774
#> alpha_j     -0.02014888 0.39763412
#> alpha_j      0.22318277 0.21847161
#> alpha_j      0.14728087 0.27057852
#> alpha_j     -0.26514652 0.14638317
#> theta_z      0.68016356 0.11510781
#> theta_z      0.68285927 0.15773444
#> theta_z      0.31701846 0.10358197
#> theta_z      0.52123769 0.10914344
#> theta_z      0.14819781 0.09200614
#> theta_z      0.51873790 0.13709521
#> theta_z     -0.31998633 0.10049164
#> theta_z      0.23601680 0.10640963
#> theta_z     -0.21613586 0.09652980
#> log_sigma -0.52205331 0.06761637
#> log_sigma 0.21851154 0.13313019
#> log_kappa -0.26761196 0.21030826
#> Maximum gradient component: 0.002233932
#>
#> $run_time
#> Time difference of 1.384631 secs

```

We can compare the true loadings (rotated to optimize comparison):

```
Lrot_cf = rotate_pca( L_cf )$L_tf
dimnames(Lrot_cf) = list( paste0("Var ", 1:nrow(Lrot_cf)),
                           paste0("Factor ", 1:ncol(Lrot_cf)) )
knitr::kable( Lrot_cf,
               digits=2, caption="Rotated true loadings")
```

Table 2: Rotated true loadings

|       | Factor 1 | Factor 2 | Factor 3 | Factor 4 | Factor 5 |
|-------|----------|----------|----------|----------|----------|
| Var 1 | -0.71    | 0.34     | 0.00     | -0.11    | -0.19    |
| Var 2 | -1.20    | -0.18    | 0.00     | -0.18    | 0.12     |
| Var 3 | 0.22     | 0.73     | -0.17    | -0.09    | 0.10     |
| Var 4 | -0.70    | 0.25     | 0.06     | 0.37     | 0.03     |
| Var 5 | 0.17     | 0.23     | 0.47     | -0.08    | 0.03     |

with the estimated loadings

```
# Extract and rotate estimated loadings
Lhat_cf = matrix( 0, nrow=n_c, ncol=2 )
Lhat_cf[lower.tri(Lhat_cf,diag=TRUE)] = as.list(out$sdrep, what="Estimate")$theta_z
Lhat_cf = rotate_pca( L_tf=Lhat_cf, order="decreasing" )$L_tf
#> Warning in sqrt(Eigen$values): NaNs produced
```

Where we can compared the estimated and true loadings matrices:

```
dimnames(Lhat_cf) = list( paste0("Var ", 1:nrow(Lhat_cf)),
                           paste0("Factor ", 1:ncol(Lhat_cf)) )
knitr::kable( Lhat_cf,
               digits=2, caption="Rotated estimated loadings" )
```

Table 3: Rotated estimated loadings

|       | Factor 1 | Factor 2 |
|-------|----------|----------|
| Var 1 | 0.64     | -0.23    |
| Var 2 | 0.82     | 0.26     |
| Var 3 | 0.19     | -0.41    |
| Var 4 | 0.57     | 0.05     |
| Var 5 | 0.07     | -0.25    |

Or we can specify the model while ensuring that residual spatial variation is also captured:

```
#
sem = "
  f1 -> 1, 11
  f1 -> 2, 12
  f1 -> 3, 13
  f1 -> 4, 14
```

```

f1 -> 5, 15
f2 -> 2, 16
f2 -> 3, 17
f2 -> 4, 18
f2 -> 5, 19
f1 <-> f1, NA, 1
f2 <-> f2, NA, 1
1 <-> 1, sd_resid
2 <-> 2, sd_resid
3 <-> 3, sd_resid
4 <-> 4, sd_resid
5 <-> 5, sd_resid
"

# fit model
out = fit( sem = sem,
  data = Data,
  formula = n ~ 0 + factor(species),
  spatial_graph = mesh,
  family = list( "obs"=tweedie() ),
  variables = c( "f1", "f2", 1:n_c ),
  data_colnames = list(spatial = c("x","y"), variable = "species", time = "time", distribution = "tweedie"),
  control = tinyVASTcontrol(quiet=TRUE, trace=0, gmrf="proj") )

# Extract and rotate estimated loadings
Lhat_cf = matrix( 0, nrow=n_c, ncol=2 )
Lhat_cf[lower.tri(Lhat_cf,diag=TRUE)] = as.list(out$sdrep, what="Estimate")$theta_z
#> Warning in Lhat_cf[lower.tri(Lhat_cf, diag = TRUE)] = as.list(out$sdrep, : number of items to replace is not a length
Lhat_cf = rotate_pca( L_tf=Lhat_cf, order="decreasing" )$L_tf
#> Warning in sqrt(Eigen$values): NaNs produced

```

Where we can again compared the estimated and true loadings matrices:

```

dimnames(Lhat_cf) = list( paste0("Var ", 1:nrow(Lhat_cf)),
  paste0("Factor ", 1:ncol(Lhat_cf)) )
knitr::kable( Lhat_cf,
  digits=2, caption="Rotated estimated loadings with full rank" )

```

Table 4: Rotated estimated loadings with full rank

|       | Factor 1 | Factor 2 |
|-------|----------|----------|
| Var 1 | 0.69     | -0.18    |
| Var 2 | 0.74     | 0.23     |
| Var 3 | 0.07     | -0.42    |
| Var 4 | 0.47     | -0.02    |
| Var 5 | 0.07     | -0.07    |