

VAST

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 the vector-autoregressive spatio-temporal component.

Spatio-temporal autoregressive model

We first explore the ability to specify a first-order autoregressive spatio-temporal process:

```
# Simulate settings
theta_xy = 0.4
n_x = n_y = 10
n_t = 15
rho = 0.8
spatial_sd = 0.5

# Simulate GMRFs
R_s = exp(-theta_xy * abs(outer(1:n_x, 1:n_y, FUN="-")))
V_ss = spatial_sd^2 * kronecker(R_s, R_s)
d = mvtnorm::rmvnorm(n_t, sigma=V_ss)

# Project through time and add mean
for( t in seq_len(n_t) ){
  if(t>1) d[t,] = rho*d[t-1,] + d[t,]
}
#d = d + 0.5

# Shape into longform data-frame and add error
Data = data.frame( expand.grid(time=1:n_t, x=1:n_x, y=1:n_y), "var"="logn", z=exp(as.vector(d)))
Data$n = tweedie::rtweedie( n=nrow(Data), mu=Data$z, phi=0.5, power=1.5 )
mean(Data$n==0)
#> [1] 0.046

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

# fit model
mytinyVAST = fit( dsem = "logn -> logn, 1, rho",
  data = Data,
  formula = n ~ 0 + factor(time),
```

```

        spatial_graph = mesh,
        family = list( "obs"=tweedie() ),
        control = tinyVASTcontrol(quiet=TRUE, trace=0) )
mytinyVAST
#> $call
#> fit(data = Data, formula = n ~ 0 + factor(time), dsem = "logn -> logn, 1, rho",
#>      family = list(obs = tweedie()), spatial_graph = mesh, control = tinyVASTcontrol(quiet = TRUE,
#>      trace = 0))
#>
#> $opt
#> $opt$par
#>      alpha_j      alpha_j      alpha_j      alpha_j      alpha_j      alpha_j      alpha_j      alpha_j
#> -0.08323603 -0.13549103 -0.10579217 -0.14499111 -0.37823867 -0.21633304 -0.41489958 -0.67168422 -0.4
#>      alpha_j      beta_z      beta_z      log_sigma      log_sigma      log_kappa
#> 0.30040122 0.81229113 0.40988915 -0.64868475 0.04394543 0.07228543
#>
#> $opt$objective
#> [1] 1717.689
#>
#> $opt$convergence
#> [1] 0
#>
#> $opt$iterations
#> [1] 77
#>
#> $opt$evaluations
#> function gradient
#>      107      77
#>
#> $opt$message
#> [1] "relative convergence (4)"
#>
#>
#> $sdrep
#> sdreport(.) result
#>
#>      Estimate Std. Error
#> alpha_j      -0.08323603 0.15196456
#> alpha_j      -0.13549103 0.18670340
#> alpha_j      -0.10579217 0.20529851
#> alpha_j      -0.14499111 0.21780791
#> alpha_j      -0.37823867 0.22691800
#> alpha_j      -0.21633304 0.23026450
#> alpha_j      -0.41489958 0.23456777
#> alpha_j      -0.67168422 0.23833635
#> alpha_j      -0.49463135 0.23869331
#> alpha_j      -0.13968722 0.23733095
#> alpha_j      0.14836185 0.23640201
#> alpha_j      -0.21516692 0.23873590
#> alpha_j      -0.20120062 0.23979214
#> alpha_j      0.16887043 0.23708655
#> alpha_j      0.30040122 0.23660035
#> beta_z      0.81229113 0.03708631
#> beta_z      0.40988915 0.03291043

```

```

#> log_sigma -0.64868475 0.05422114
#> log_sigma  0.04394543 0.07275797
#> log_kappa  0.07228543 0.10755269
#> Maximum gradient component: 0.006334975
#>
#> $run_time
#> Time difference of 21.21378 secs

```

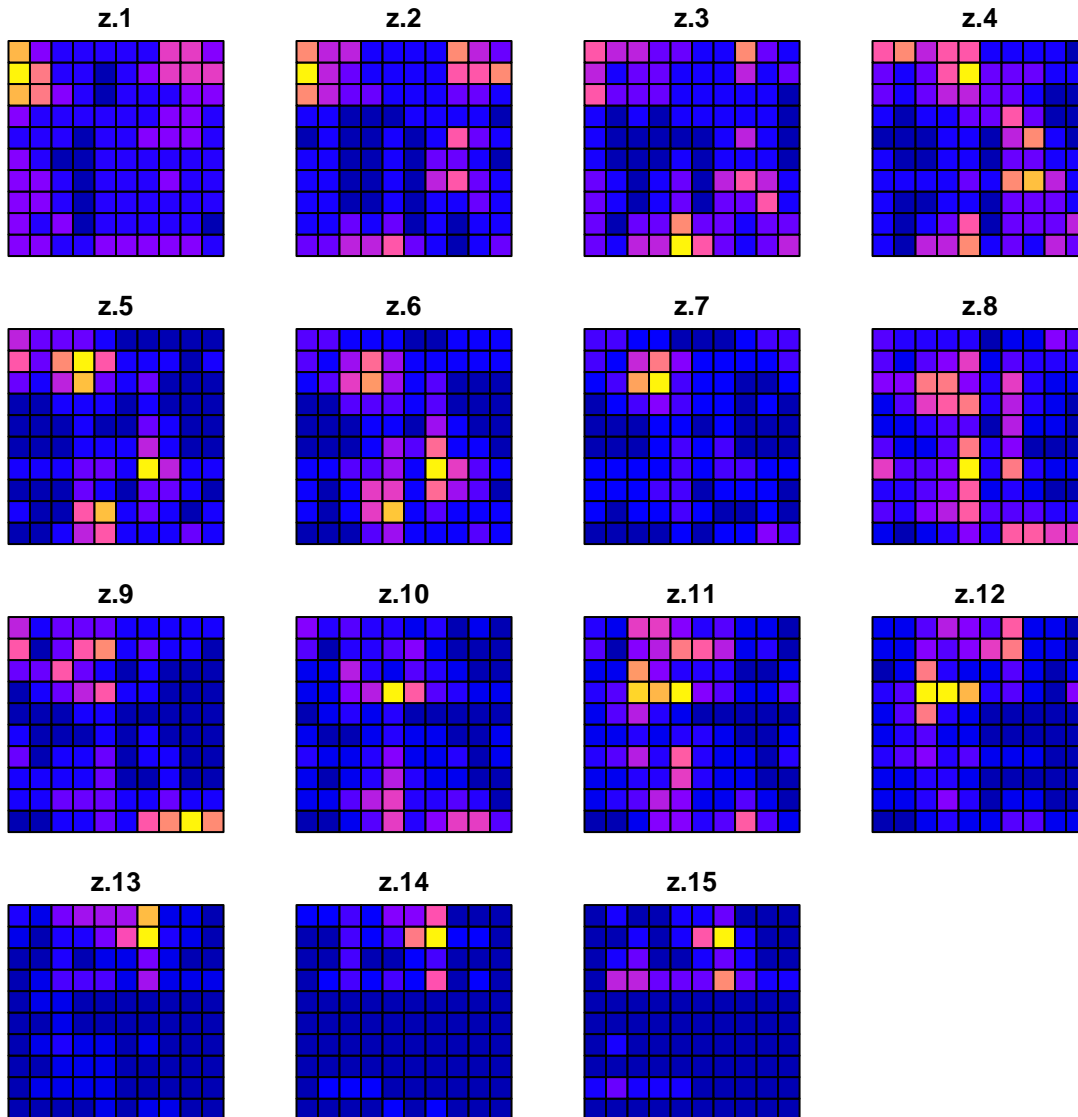
The estimated values for **beta_z** then correspond to the simulated value for **rho** and **spatial_sd**.

We can compare the true densities:

```

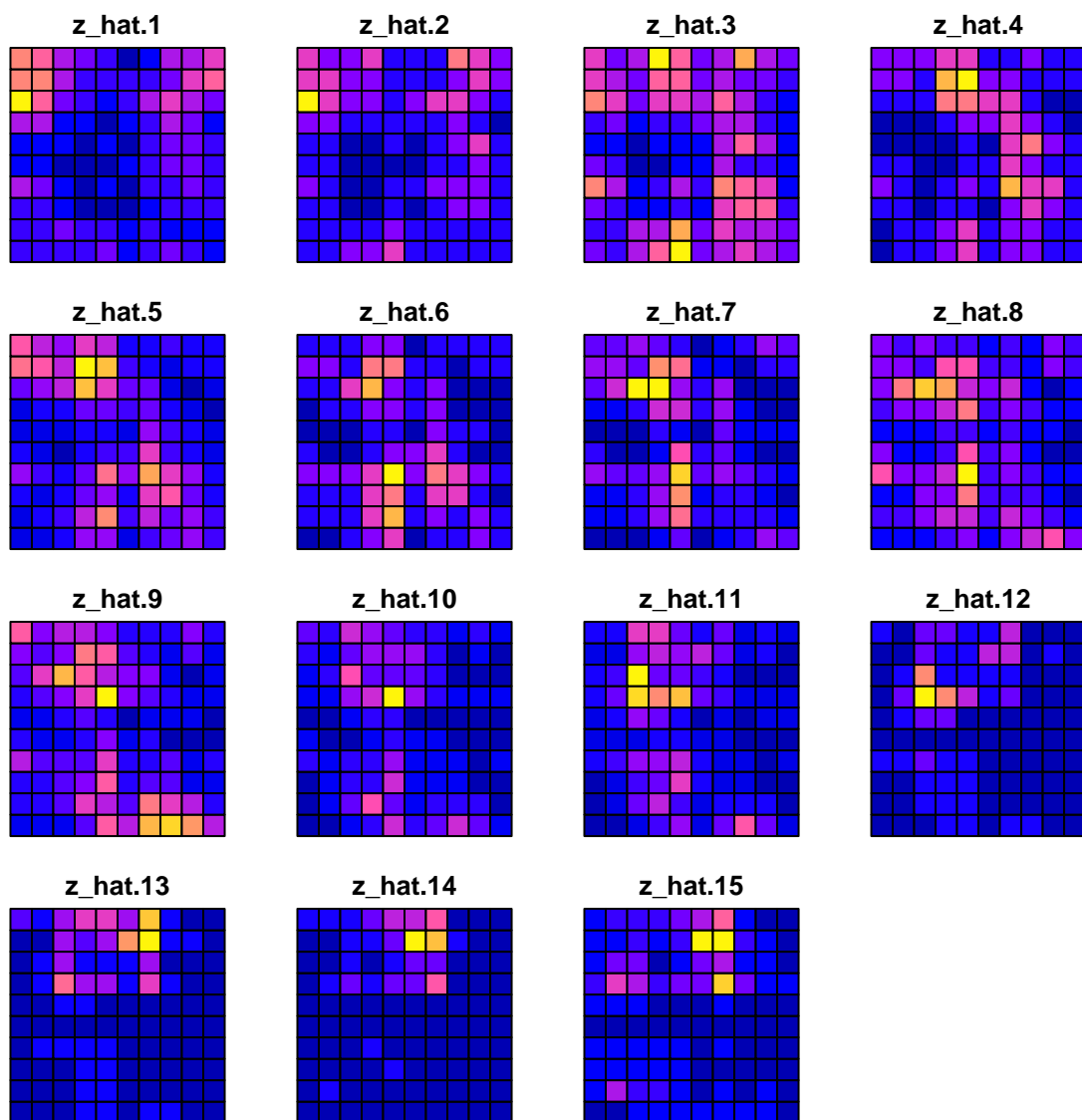
library(sf)
data_wide = reshape( Data[,c('x','y','time','z')],
                      direction = "wide", idvar = c('x','y'), timevar = "time")
sf_data = st_as_sf( data_wide, coords=c("x","y"))
sf_grid = sf::st_make_grid( sf_data )
sf_plot = st_sf(sf_grid, st_drop_geometry(sf_data) )
plot(sf_plot, max.plot=n_t )

```



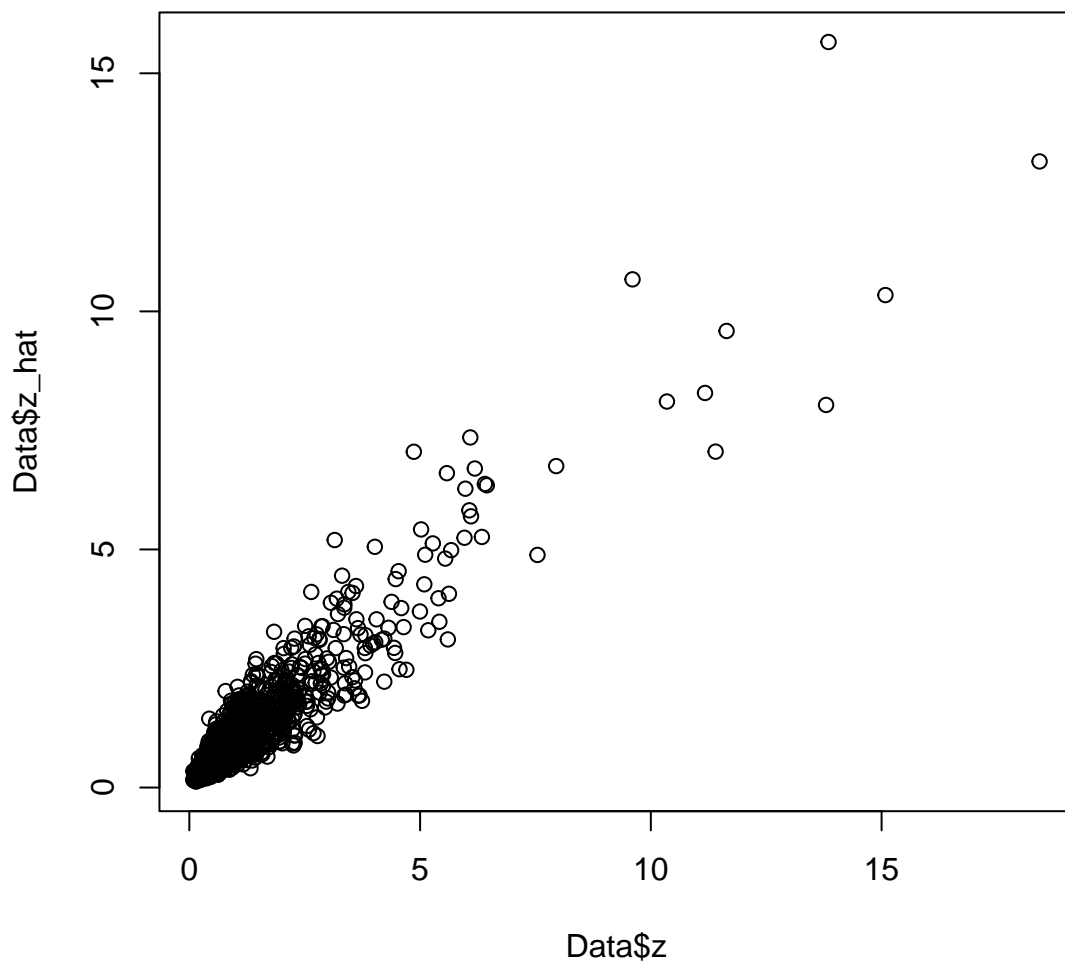
with the estimated densities:

```
Data$z_hat = predict(mytinyVAST)
data_wide = reshape( Data[,c('x','y','time','z_hat')],
                     direction = "wide", idvar = c('x','y'), timevar = "time")
sf_data = st_as_sf( data_wide, coords=c("x","y"))
sf_plot = st_sf(sf_grid, st_drop_geometry(sf_data) )
plot(sf_plot, max.plot=n_t )
```



where a scatterplot shows that they are highly correlated:

```
plot( x=Data$z, y=Data$z_hat )
```



We can then calculate the area-weighted total abundance and compare it with its true value:

```
# Predicted sample-weighted total
(Est = sapply( seq_len(n_t),
  FUN=\(t) integrate_output(mytinyVAST, newdata=subset(Data,time==t)) ))
#>           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
#> Estimate    97.164903  96.643634  98.362457 101.517620  84.760587  97.538111  77.520820  59.56
#> Std. Error    7.194683   7.216494   7.309312   7.572241   6.643419   7.406226   6.207919   5.09
#> Est. (bias.correct) 102.324275 102.850496 105.043003 108.373177  90.604659 104.258111  83.102278  64.06
#> Std. (bias.correct)      NA      NA      NA      NA      NA      NA      NA
#>           [,15]
#> Estimate    187.86973
#> Std. Error    12.88189
#> Est. (bias.correct) 200.54754
#> Std. (bias.correct)      NA

# True (latent) sample-weighted total
```

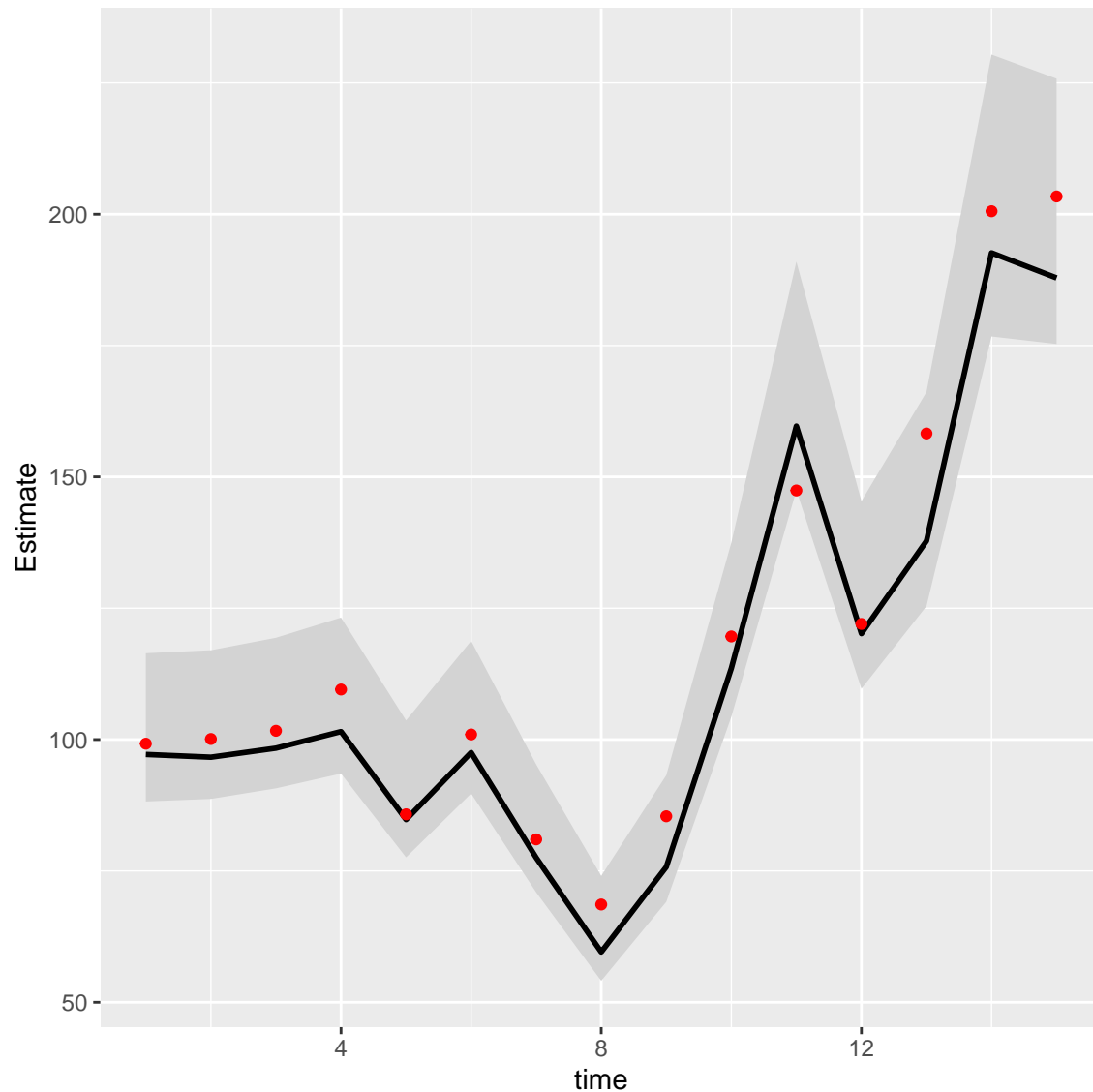
```

(True = tapply( Data$z, INDEX=Data$time, FUN=sum ))
#>      1      2      3      4      5      6      7      8      9     10
#> 99.21643 100.10603 101.66846 109.52622 85.76973 100.97116 80.99847 68.60738 85.39974 119.62380

#
Index = data.frame( time=seq_len(n_t), t(Est), True )
Index$low = Index[, 'Est...bias.correct.'] - 1.96*Index[, 'Std..Error']
Index$high = Index[, 'Est...bias.correct.'] + 1.96*Index[, 'Std..Error']

#
library(ggplot2)
ggplot(Index, aes(time, Estimate)) +
  geom_ribbon(aes(ymin = low,
                 ymax = high),      # shadowing cnf intervals
             fill = "lightgrey") +
  geom_line( color = "black",
             linewidth = 1) +
  geom_point( aes(time, True), color = "red" )

```



Next, we compare this against the current version of VAST

```
library(VAST)
settings = make_settings( purpose="index3",
                          n_x = n_x*n_y,
                          Region = "Other",
                          bias.correct = FALSE,
                          use_anisotropy = FALSE )
settings$FieldConfig['Epsilon','Component_1'] = 0
settings$FieldConfig['Omega',] = 0
settings$RhoConfig['Epsilon2'] = 4
settings$RhoConfig['Beta1'] = 3
settings$ObsModel = c(10,2)

myVAST = fit_model( settings=settings,
                    Lat_i = Data[, 'y'],
                    Lon_i = Data[, 'x'],
```



```

t_i = Data[, 'time'],
b_i = Data[, 'n'],
a_i = rep(1, nrow(Data)),
observations_LL = cbind(Lat=Data[, 'y'], Lon=Data[, 'x']),
grid_dim_km = c(100, 100),
newtonsteps = 0,
loopnum = 1,
control = list(eval.max=100, iter.max=100, trace=0) )
#> Warning: The `returnclass` argument of `ne_download()` sp as of rnaturalearth 1.0.0.
#> i Please use `sf` objects with {rnaturalearth}, support for Spatial objects (sp) will be removed in
#> i The deprecated feature was likely used in the FishStatsUtils package.
#> Please report the issue to the authors.
#> This warning is displayed once every 8 hours.
#> Call `lifecycle::last_lifecycle_warnings()` to see where this warning was generated.
#> Warning in FishStatsUtils:::inla.barrier.fem.copy(mesh = anisotropic_mesh, : Please install the `INL
#> which contains an implementation that runs faster!
#>      Component_1 Component_2
#> Omega          -1          -1
#> Epsilon         -1          -2
#> Beta            -2          -2
#> Epsilon_time    -3          -3
#> Eta1 Eta2
#>  -1  -1
#>      Coefficient_name Number_of_coefficients Type
#> 1      beta1_ft          1 Fixed
#> 2      beta2_ft          15 Fixed
#> 3  Epsilon_rho2_f          1 Fixed
#> 4      L_epsilon2_z          1 Fixed
#> 5      logkappa2          1 Fixed
#> 6      logSigmaM          1 Fixed
#> 7 Epsiloninput2_sf      2040 Random
#> 0: 2105.7266: 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000 0.00000
#> 1: 1993.4850: 0.248169 -0.00621313 -0.00940275 -0.00474146 -0.00488678 -0.0304006 -0.0100244 -
#> 2: 1815.3664: -0.625674 -0.0250910 -0.0263651 -0.0154445 -0.0135640 -0.0640075 -0.0194314 -0.0
#> 3: 1787.0493: -0.401654 0.00970346 -0.0193647 -0.00730560 -0.00492316 -0.0744611 -0.00503150 -
#> 4: 1777.8725: -0.446246 0.0170642 -0.0177169 -0.00500788 -0.00228127 -0.0780425 -0.000709483 -
#> 5: 1769.4163: -0.519117 0.0790811 0.00101357 0.0150763 0.0200457 -0.0994408 0.0309285 -0.06803
#> 6: 1760.3172: -0.511935 0.157190 0.0320130 0.0398107 0.0442341 -0.117800 0.0594775 -0.0745811
#> 7: 1750.8485: -0.566256 0.242491 0.0751094 0.0723315 0.0724627 -0.130056 0.0861515 -0.0818699
#> 8: 1747.1048: -0.570896 0.297687 0.124667 0.106758 0.0886785 -0.116014 0.0852856 -0.0872344 -0
#> 9: 1744.9699: -0.509053 0.311068 0.139401 0.117785 0.0947485 -0.112493 0.0849978 -0.0893639 -0
#> 10: 1743.1222: -0.512816 0.321511 0.166698 0.139649 0.108323 -0.104951 0.0823444 -0.0952418 -0.
#> 11: 1742.7069: -0.578092 0.317902 0.191272 0.161171 0.123579 -0.0949179 0.0755250 -0.102868 -0.
#> 12: 1741.4359: -0.553625 0.350576 0.228002 0.198994 0.151349 -0.0855427 0.0768123 -0.112226 -0.
#> 13: 1741.3951: -0.549230 0.354115 0.230379 0.201585 0.153643 -0.0848625 0.0779499 -0.112529 -0.
#> 14: 1741.3352: -0.551012 0.357402 0.232642 0.204355 0.156205 -0.0837881 0.0792783 -0.112707 -0.
#> 15: 1741.2774: -0.550206 0.360788 0.235473 0.207574 0.159120 -0.0822459 0.0806568 -0.112809 -0.
#> 16: 1741.2267: -0.551500 0.363838 0.238499 0.210823 0.161987 -0.0806066 0.0819352 -0.112867 -0.
#> 17: 1741.1721: -0.550622 0.366481 0.241688 0.214111 0.164885 -0.0786655 0.0831737 -0.112753 -0.
#> 18: 1741.1236: -0.551920 0.368996 0.244873 0.217411 0.167764 -0.0767614 0.0844078 -0.112601 -0.
#> 19: 1741.0724: -0.551195 0.371249 0.248032 0.220698 0.170664 -0.0746785 0.0856768 -0.112263 -0.
#> 20: 1741.0260: -0.552449 0.373520 0.251087 0.223967 0.173556 -0.0726804 0.0870056 -0.111886 -0.
#> 21: 1740.9777: -0.551772 0.375592 0.254042 0.227169 0.176446 -0.0705493 0.0884075 -0.111306 -0.

```

#> 22:	1740.9333:	-0.552979	0.377691	0.256908	0.230329	0.179311	-0.0684846	0.0898717	-0.110681	-0.110681	-0.110681
#> 23:	1740.8874:	-0.552354	0.379597	0.259661	0.233382	0.182143	-0.0663000	0.0914165	-0.109843	-0.109843	-0.109843
#> 24:	1740.8448:	-0.553529	0.381533	0.262347	0.236393	0.184943	-0.0641761	0.0930184	-0.108966	-0.108966	-0.108966
#> 25:	1740.8009:	-0.552945	0.383290	0.264911	0.239281	0.187690	-0.0619540	0.0947033	-0.107879	-0.107879	-0.107879
#> 26:	1740.7599:	-0.554087	0.385088	0.267422	0.242138	0.190408	-0.0597943	0.0964357	-0.106769	-0.106769	-0.106769
#> 27:	1740.7179:	-0.553533	0.386717	0.269810	0.244868	0.193066	-0.0575534	0.0982451	-0.105458	-0.105458	-0.105458
#> 28:	1740.6785:	-0.554646	0.388394	0.272165	0.247582	0.195703	-0.0553732	0.100088	-0.104144	-0.104144	-0.104144
#> 29:	1740.6381:	-0.554116	0.389913	0.274398	0.250169	0.198274	-0.0531233	0.101999	-0.102642	-0.102642	-0.102642
#> 30:	1740.5999:	-0.555202	0.391487	0.276618	0.252755	0.200835	-0.0509313	0.103932	-0.101154	-0.101154	-0.101154
#> 31:	1740.5609:	-0.554691	0.392909	0.278720	0.255217	0.203329	-0.0486780	0.105925	-0.0994921	-0.0994921	-0.0994921
#> 32:	1740.5240:	-0.555753	0.394396	0.280824	0.257695	0.205823	-0.0464791	0.107931	-0.0978588	-0.0978588	-0.0978588
#> 33:	1740.4863:	-0.555260	0.395736	0.282815	0.260052	0.208250	-0.0442245	0.109991	-0.0960631	-0.0960631	-0.0960631
#> 34:	1740.4504:	-0.556300	0.397148	0.284822	0.262437	0.210687	-0.0420205	0.112057	-0.0943084	-0.0943084	-0.0943084
#> 35:	1740.4138:	-0.555822	0.398418	0.286719	0.264704	0.213058	-0.0397646	0.114172	-0.0924001	-0.0924001	-0.0924001
#> 36:	1740.3790:	-0.556842	0.399766	0.288644	0.267012	0.215447	-0.0375560	0.116289	-0.0905430	-0.0905430	-0.0905430
#> 37:	1740.3433:	-0.556378	0.400974	0.290460	0.269203	0.217770	-0.0352983	0.118450	-0.0885390	-0.0885390	-0.0885390
#> 38:	1740.3094:	-0.557382	0.402268	0.292316	0.271446	0.220119	-0.0330849	0.120611	-0.0865946	-0.0865946	-0.0865946
#> 39:	1740.2747:	-0.556930	0.403424	0.294064	0.273572	0.222402	-0.0308244	0.122813	-0.0845081	-0.0845081	-0.0845081
#> 40:	1740.2416:	-0.557918	0.404671	0.295860	0.275760	0.224718	-0.0286058	0.125012	-0.0824886	-0.0824886	-0.0824886
#> 41:	1740.2077:	-0.557478	0.405780	0.297549	0.277831	0.226967	-0.0263417	0.127249	-0.0803302	-0.0803302	-0.0803302
#> 42:	1740.1754:	-0.558453	0.406989	0.299295	0.279972	0.229255	-0.0241174	0.129481	-0.0782454	-0.0782454	-0.0782454
#> 43:	1740.1423:	-0.558024	0.408057	0.300933	0.281996	0.231475	-0.0218490	0.131750	-0.0760237	-0.0760237	-0.0760237
#> 44:	1740.1107:	-0.558986	0.409232	0.302637	0.284098	0.233741	-0.0196188	0.134013	-0.0738817	-0.0738817	-0.0738817
#> 45:	1739.1745:	-0.567381	0.435950	0.399872	0.419534	0.401618	0.196149	0.349401	0.173937	-0.0615	-0.0615
#> 46:	1738.8279:	-0.596524	0.471652	0.397182	0.441269	0.441486	0.225559	0.413372	0.232199	0.00229	0.00229
#> 47:	1738.5935:	-0.588631	0.486598	0.424664	0.445106	0.443186	0.233576	0.433439	0.257339	-0.0001	-0.0001
#> 48:	1738.5330:	-0.588773	0.482571	0.456978	0.479450	0.446830	0.251282	0.408587	0.228845	-0.0112	-0.0112
#> 49:	1738.4986:	-0.586549	0.503767	0.458645	0.501052	0.483951	0.237362	0.397869	0.199913	-0.0305	-0.0305
#> 50:	1738.4741:	-0.588433	0.526758	0.465209	0.502090	0.498809	0.229918	0.397908	0.194546	-0.0755	-0.0755
#> 51:	1738.4317:	-0.595944	0.528783	0.468489	0.500110	0.482969	0.237850	0.406101	0.192359	-0.0929	-0.0929
#> 52:	1738.3873:	-0.593420	0.521139	0.470568	0.496121	0.466359	0.237208	0.408074	0.179406	-0.0851	-0.0851
#> 53:	1738.3670:	-0.593640	0.520784	0.470148	0.495289	0.468669	0.236030	0.403759	0.182930	-0.0831	-0.0831
#> 54:	1738.3647:	-0.592189	0.520681	0.470006	0.495198	0.469040	0.235710	0.402975	0.183604	-0.0827	-0.0827
#> 55:	1738.3536:	-0.592538	0.519602	0.469976	0.494577	0.469653	0.234602	0.401502	0.185091	-0.0809	-0.0809
#> 56:	1738.3495:	-0.591602	0.518839	0.469263	0.494263	0.469916	0.233437	0.400140	0.186819	-0.0789	-0.0789
#> 57:	1738.3457:	-0.591405	0.518700	0.468028	0.494016	0.469236	0.232796	0.400074	0.188030	-0.0771	-0.0771
#> 58:	1738.3431:	-0.590947	0.518128	0.467835	0.492128	0.469429	0.233065	0.400265	0.189199	-0.0761	-0.0761
#> 59:	1738.3414:	-0.590732	0.516841	0.466226	0.493451	0.469814	0.232361	0.398789	0.191112	-0.0739	-0.0739
#> 60:	1738.3404:	-0.590639	0.517947	0.466264	0.493252	0.469227	0.230588	0.400163	0.190894	-0.0716	-0.0716
#> 61:	1738.3396:	-0.590399	0.518399	0.467536	0.490949	0.469168	0.231677	0.399523	0.192403	-0.0718	-0.0718
#> 62:	1738.3393:	-0.590041	0.518003	0.467475	0.490879	0.468970	0.232448	0.398910	0.193122	-0.0722	-0.0722
#> 63:	1738.3384:	-0.589853	0.516711	0.466863	0.491482	0.469107	0.232961	0.398825	0.192952	-0.0726	-0.0726
#> 64:	1738.3381:	-0.589898	0.516290	0.465766	0.491697	0.470084	0.231935	0.398781	0.192302	-0.0719	-0.0719
#> 65:	1738.3378:	-0.590047	0.517086	0.465802	0.491283	0.469494	0.231363	0.397506	0.193365	-0.0711	-0.0711
#> 66:	1738.3377:	-0.589550	0.516628	0.465973	0.491706	0.468490	0.230889	0.397833	0.193791	-0.0706	-0.0706
#> 67:	1738.3376:	-0.589612	0.515421	0.465693	0.491652	0.468542	0.231085	0.398274	0.193531	-0.0711	-0.0711
#> 68:	1738.3374:	-0.589379	0.515397	0.465269	0.491285	0.468871	0.231574	0.398032	0.193246	-0.0715	-0.0715
#> 69:	1738.3374:	-0.589387	0.516151	0.465173	0.490950	0.468610	0.231381	0.397811	0.193109	-0.0708	-0.0708
#> 70:	1738.3373:	-0.589579	0.515992	0.465411	0.490953	0.468609	0.230659	0.397439	0.193530	-0.0708	-0.0708
#> 71:	1738.3373:	-0.589571	0.515993	0.465395	0.490960	0.468613	0.230653	0.397427	0.193531	-0.0708	-0.0708
#> 72:	1738.3373:	-0.589531	0.515994	0.465366	0.490973	0.468620	0.230644	0.397406	0.193533	-0.0707	-0.0707
#> 73:	1738.3373:	-0.589531	0.515991	0.465339	0.490982	0.468627	0.230636	0.397387	0.193533	-0.0707	-0.0707
#> 74:	1738.3373:	-0.589505	0.515976	0.465305	0.490981	0.468633	0.230638	0.397365	0.193528	-0.0707	-0.0707

```

#> 75: 1738.3373: -0.589501 0.515963 0.465270 0.490982 0.468639 0.230634 0.397341 0.193523 -0.0707
#> 76: 1738.3373: -0.589476 0.515942 0.465232 0.490970 0.468644 0.230632 0.397311 0.193511 -0.0707
#> 77: 1738.3373: -0.589470 0.515924 0.465196 0.490962 0.468644 0.230618 0.397280 0.193502 -0.0707
#> 78: 1738.3373: -0.589450 0.515903 0.465165 0.490947 0.468632 0.230592 0.397249 0.193491 -0.0707
#> 79: 1738.3373: -0.589448 0.515887 0.465135 0.490939 0.468618 0.230562 0.397221 0.193484 -0.0707
#> 80: 1738.3373: -0.589432 0.515873 0.465106 0.490918 0.468596 0.230532 0.397195 0.193472 -0.0707
#> 81: 1738.3373: -0.589431 0.515857 0.465076 0.490910 0.468581 0.230503 0.397167 0.193468 -0.0707
#> 82: 1738.3373: -0.589416 0.515833 0.465055 0.490892 0.468558 0.230471 0.397140 0.193461 -0.0707
#> 83: 1738.3373: -0.589414 0.515832 0.465019 0.490888 0.468554 0.230450 0.397115 0.193455 -0.0706
#> 84: 1738.3373: -0.589401 0.515828 0.464989 0.490864 0.468537 0.230431 0.397087 0.193438 -0.0706
#> 85: 1738.3373: -0.589401 0.515803 0.464977 0.490865 0.468524 0.230402 0.397068 0.193441 -0.0706
#> 86: 1738.3373: -0.589387 0.515777 0.464969 0.490856 0.468503 0.230363 0.397056 0.193445 -0.0706
#> 87: 1738.3373: -0.589385 0.515781 0.464934 0.490839 0.468490 0.230348 0.397029 0.193431 -0.0706
#> 88: 1738.3373: -0.589374 0.515769 0.464908 0.490825 0.468485 0.230335 0.396999 0.193416 -0.0706
#> 89: 1738.3373: -0.589373 0.515738 0.464904 0.490800 0.468466 0.230302 0.396980 0.193406 -0.0706
#> 90: 1738.3373: -0.589346 0.515737 0.464863 0.490812 0.468459 0.230282 0.396958 0.193412 -0.0706
#> 91: 1738.3373: -0.589355 0.515745 0.464814 0.490819 0.468427 0.230274 0.396935 0.193411 -0.0706
#> 92: 1738.3373: -0.589353 0.515721 0.464819 0.490766 0.468436 0.230252 0.396916 0.193382 -0.0706
#> 93: 1738.3373: -0.589346 0.515700 0.464810 0.490745 0.468409 0.230219 0.396887 0.193371 -0.0706
#> 94: 1738.3373: -0.589337 0.515689 0.464780 0.490748 0.468364 0.230185 0.396866 0.193376 -0.0706
#> 95: 1738.3373: -0.589338 0.515677 0.464768 0.490731 0.468346 0.230141 0.396879 0.193382 -0.0706
#> 96: 1738.3373: -0.589341 0.515672 0.464734 0.490714 0.468326 0.230152 0.396840 0.193351 -0.0706
#> 97: 1738.3373: -0.589333 0.515663 0.464710 0.490699 0.468307 0.230136 0.396799 0.193330 -0.0706
#> 98: 1738.3373: -0.589319 0.515645 0.464699 0.490673 0.468292 0.230092 0.396783 0.193323 -0.0706
#> 99: 1738.3373: -0.589319 0.515623 0.464680 0.490645 0.468260 0.230067 0.396786 0.193314 -0.0706
#> 100: 1738.3373: -0.589335 0.515616 0.464667 0.490640 0.468249 0.230043 0.396745 0.193293 -0.0706
#> 101: 1738.3373: -0.589315 0.515603 0.464638 0.490619 0.468221 0.230020 0.396728 0.193274 -0.0707
#> 102: 1738.3373: -0.589305 0.515592 0.464605 0.490574 0.468212 0.230019 0.396733 0.193271 -0.0706
#> 103: 1738.3373: -0.589304 0.515589 0.464616 0.490574 0.468193 0.229976 0.396692 0.193245 -0.0706
#> 104: 1738.3373: -0.589304 0.515559 0.464589 0.490557 0.468168 0.229960 0.396668 0.193243 -0.0707
#> 105: 1738.3373: -0.589302 0.515566 0.464581 0.490553 0.468152 0.229949 0.396668 0.193226 -0.0707
#> 106: 1738.3373: -0.589302 0.515566 0.464581 0.490553 0.468152 0.229949 0.396668 0.193226 -0.0707

```

```
myVAST
```

```
#> fit_model(.) result
```

```
#> $par
```

```

#>      beta1_ft      beta2_ft      beta2_ft      beta2_ft      beta2_ft      beta2_ft      beta
#> -0.58930073    0.51556566    0.46458140    0.49054922    0.46815477    0.22995173    0.3966
#>      beta2_ft      beta2_ft      beta2_ft      beta2_ft      beta2_ft      L_epsilon2_z      logka
#>      0.77135923    0.40831816    0.42839452    0.79998255    0.91170571    0.49266124    -4.3006
#>

```

```
#> $objective
```

```
#> [1] 1738.337
```

```
#>
```

```
#> $iterations
```

```
#> [1] 6
```

```
#>
```

```
#> $evaluations
```

```
#> function gradient
```

```
#>      12      7
```

```
#>
```

```
#> $time_for_MLE
```

```
#> Time difference of 1.115946 secs
```

```
#>
```

```

#> $max_gradient
#> [1] 0.0005701444
#>
#> $Convergence_check
#> [1] "The model is likely not converged"
#>
#> $number_of_coefficients
#> Total Fixed Random
#> 2060 20 2040
#>
#> $AIC
#> [1] 3516.675
#>
#> $diagnostics
#> Param starting_value Lower MLE Upper final_gradient
#> 1 beta1_ft -0.58930212 -Inf -0.58930073 Inf -2.542160e-04
#> 2 beta2_ft 0.51556575 -Inf 0.51556566 Inf 7.802298e-06
#> 3 beta2_ft 0.46458086 -Inf 0.46458140 Inf -1.581373e-05
#> 4 beta2_ft 0.49055256 -Inf 0.49054922 Inf 2.253090e-04
#> 5 beta2_ft 0.46815191 -Inf 0.46815477 Inf -2.143175e-04
#> 6 beta2_ft 0.22994939 -Inf 0.22995173 Inf -1.957531e-04
#> 7 beta2_ft 0.39666758 -Inf 0.39666652 Inf 7.238230e-05
#> 8 beta2_ft 0.19322638 -Inf 0.19322492 Inf 1.270660e-04
#> 9 beta2_ft -0.07072351 -Inf -0.07072447 Inf 8.178894e-05
#> 10 beta2_ft 0.11680232 -Inf 0.11680344 Inf -9.221727e-05
#> 11 beta2_ft 0.47100947 -Inf 0.47101097 Inf -1.199564e-04
#> 12 beta2_ft 0.77135987 -Inf 0.77135923 Inf 3.451023e-05
#> 13 beta2_ft 0.40831868 -Inf 0.40831816 Inf 3.772265e-05
#> 14 beta2_ft 0.42839440 -Inf 0.42839452 Inf -2.666228e-05
#> 15 beta2_ft 0.79997988 -Inf 0.79998255 Inf -1.683173e-04
#> 16 beta2_ft 0.91170844 -Inf 0.91170571 Inf 1.977504e-04
#> 17 L_epsilon2_z 0.49266062 -Inf 0.49266124 Inf -3.371048e-04
#> 18 logkappa2 -4.30062170 -6.214608 -4.30062277 -3.565449 1.075603e-04
#> 19 Epsilon_rho2_f 0.85035618 -0.990000 0.85035927 0.990000 -5.701444e-04
#> 20 logSigmaM 0.10422369 -Inf 0.10422305 10.000000 8.758814e-05
#>
#> $SD
#> sdreport(.) result
#> Estimate Std. Error
#> beta1_ft -0.58930073 0.05080467
#> beta2_ft 0.51556566 0.14414517
#> beta2_ft 0.46458140 0.17371753
#> beta2_ft 0.49054922 0.19200615
#> beta2_ft 0.46815477 0.20433941
#> beta2_ft 0.22995173 0.21476798
#> beta2_ft 0.39666652 0.21934844
#> beta2_ft 0.19322492 0.22481460
#> beta2_ft -0.07072447 0.22973074
#> beta2_ft 0.11680344 0.23059098
#> beta2_ft 0.47101097 0.22964742
#> beta2_ft 0.77135923 0.22895294
#> beta2_ft 0.40831816 0.23199190
#> beta2_ft 0.42839452 0.23305156

```

```
#> beta2_ft      0.79998255 0.23091833
#> beta2_ft      0.91170571 0.23072494
#> L_epsilon2_z  0.49266124 0.04727208
#> logkappa2     -4.30062277 0.13652887
#> Epsilon_rho2_f 0.85035927 0.03527714
#> logSigmaM     0.10422305 0.07108160
#> Maximum gradient component: 0.0005701444
#>
#> $time_for_sdreport
#> Time difference of 4.639605 secs
#>
#> $time_for_run
#> Time difference of 21.52118 secs
```

Or with sdmTMB

```
library(sdmTMB)
mesh = make_mesh(Data, c("x","y"), n_knots=n_x*n_y )

start_time = Sys.time()
mysdmTMB = sdmTMB(
  formula = n ~ 0 + factor(time),
  data = Data,
  mesh = mesh,
  spatial = "off",
  spatiotemporal = "ar1",
  time = "time",
  family = tweedie()
)
sdmTMBtime = Sys.time() - start_time
```

The models all have similar runtimes

```
Times = c( "tinyVAST" = mytinyVAST$run_time,
           "VAST" = myVAST$total_time,
           "sdmTMB" = sdmTMBtime )
knitr::kable( cbind("run times (sec.)"=Times), digits=1)
```

run times (sec.)	
tinyVAST	21.2
VAST	23.3
sdmTMB	20.5

Delta models

We can also fit these data using a delta model

```
# fit model
mydelta = fit( data = Data,
```

```

        formula = n ~ 1,
        delta_formula = ~ 0 + factor(time),
        delta_dsem = "logn -> logn, 1, rho",
        family = list( "obs"=independent_delta() ),
        spatial_graph = mesh,
        control = tinyVASTcontrol(quiet=TRUE, trace=0) )

mydelta
#> $call
#> fit(data = Data, formula = n ~ 1, delta_formula = ~0 + factor(time),
#>      delta_dsem = "logn -> logn, 1, rho", family = list(obs = independent_delta()),
#>      spatial_graph = mesh, control = tinyVASTcontrol(quiet = TRUE,
#>      trace = 0))
#>
#> $opt
#> $opt$par
#>      alpha_j      alpha2_j      alpha2_j      alpha2_j      alpha2_j      alpha2_j      alpha2_j      alpha2_j
#> 3.032022e+00 4.467177e-01 4.113254e-01 5.053690e-01 3.694708e-01 7.254509e-02 3.336953e-01 1
#>      alpha2_j      alpha2_j      alpha2_j      alpha2_j      beta2_z      beta2_z      log_sigma
#> 5.325019e-01 2.865528e-01 7.913769e-01 9.836626e-01 1.012746e+00 8.306752e-07 2.565052e-01
#>
#> $opt$objective
#> [1] 2069.593
#>
#> $opt$convergence
#> [1] 0
#>
#> $opt$iterations
#> [1] 88
#>
#> $opt$evaluations
#> function gradient
#>      137      89
#>
#> $opt$message
#> [1] "relative convergence (4)"
#>
#>
#> $sdrep
#> sdreport(.) result
#> Warning in sqrt(diag(object$cov.fixed)): NaNs produced
#>      Estimate Std. Error
#> alpha_j      3.032022e+00 0.12325402
#> alpha2_j      4.467177e-01 0.13555038
#> alpha2_j      4.113254e-01 0.13488713
#> alpha2_j      5.053690e-01 0.13488702
#> alpha2_j      3.694708e-01 0.13423421
#> alpha2_j      7.254509e-02 0.13423422
#> alpha2_j      3.336953e-01 0.13488712
#> alpha2_j      1.252914e-01 0.13622434
#> alpha2_j     -1.360709e-01 0.13555039
#> alpha2_j      2.348312e-01 0.14050758
#> alpha2_j      4.421359e-01 0.13690899
#> alpha2_j      7.414033e-01 0.13555050

```



```

#> alpha2_j    5.325019e-01 0.13690900
#> alpha2_j    2.865528e-01 0.13903204
#> alpha2_j    7.913769e-01 0.13488708
#> alpha2_j    9.836626e-01 0.1355026
#> beta2_z     1.012746e+00      NaN
#> beta2_z     8.306752e-07      NaN
#> log_sigma   2.565052e-01 0.01869238
#> Warning:
#> Hessian of fixed effects was not positive definite.
#> Maximum gradient component: 0.006397981
#>
#> $run_time
#> Time difference of 0.1379099 secs

```

Bivariate spatio-temporal autoregressive model

We next highlight how to specify a bivariate spatio-temporal model with a cross-lagged (vector autoregressive) interaction.

```

# Simulate settings
theta_xy = 0.2
n_x = n_y = 10
n_t = 20
B = rbind( c( 0.5, -0.25),
            c(-0.1,  0.50) )

# Simulate GMRFs
R = exp(-theta_xy * abs(outer(1:n_x, 1:n_y, FUN="-"))) )
d1 = mvtnorm::rmvnorm(n_t, sigma=0.2*kronecker(R,R) )
d2 = mvtnorm::rmvnorm(n_t, sigma=0.2*kronecker(R,R) )
d = abind::abind( d1, d2, along=3 )

# Project through time and add mean
for( t in seq_len(n_t) ){
  if(t>1) d[t,,] = t*(B%*%t(d[t-1,,])) + d[t,,]
}

# Shape into longform data-frame and add error
Data = data.frame( expand.grid(time=1:n_t, x=1:n_x, y=1:n_y, "var"=c("d1","d2")), z=exp(as.vector(d)))
Data$z = tweedie::rtweedie( n=nrow(Data), mu=Data$z, phi=0.5, power=1.5 )

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

# Define DSEM
dsem = "
  d1 -> d1, 1, b11
  d2 -> d2, 1, b22
  d2 -> d1, 1, b21
  d1 -> d2, 1, b12
  d1 <-> d1, 0, var1
  d2 <-> d2, 0, var1

```

```

"

# fit model
out = fit( dsem = dsem,
           data = Data,
           formula = n ~ 0 + var,
           spatial_graph = mesh,
           family = list( "obs"=tweedie() ),
           control = tinyVASTcontrol(quiet=TRUE, trace=0) )

out
#> $call
#> fit(data = Data, formula = n ~ 0 + var, dsem = dsem, family = list(obs = tweedie()),
#>      spatial_graph = mesh, control = tinyVASTcontrol(quiet = TRUE,
#>      trace = 0))
#>
#> $opt
#> $opt$par
#>      alpha_j      alpha_j      beta_z      beta_z      beta_z      beta_z      beta_z      log_si
#> -0.090128407 -0.002000396  0.509529381  0.529236408 -0.200418878 -0.117205373  0.294319080 -0.646266
#>
#> $opt$objective
#> [1] 4365.006
#>
#> $opt$convergence
#> [1] 0
#>
#> $opt$iterations
#> [1] 52
#>
#> $opt$evaluations
#> function gradient
#>      66      53
#>
#> $opt$message
#> [1] "relative convergence (4)"
#>
#>
#> $sdrep
#> sdreport(.) result
#>
#>      Estimate Std. Error
#> alpha_j      -0.090128407 0.097711149
#> alpha_j      -0.002000396 0.09611298
#> beta_z        0.509529381 0.07886506
#> beta_z        0.529236408 0.07336727
#> beta_z       -0.200418878 0.08304602
#> beta_z       -0.117205373 0.07264403
#> beta_z        0.294319080 0.01800602
#> log_sigma    -0.646266087 0.02660900
#> log_sigma     0.012846251 0.04964136
#> log_kappa    -0.669057066 0.09746707
#> Maximum gradient component: 0.004503716
#>
#> $run_time

```



```
#> Time difference of 2.192311 mins
```

The values for `beta_z` again correspond to the specified value for interaction-matrix B

We can again calculate the area-weighted total abundance and compare it with its true value:

```
# Predicted sample-weighted total
Est1 = sapply( seq_len(n_t), FUN=\(t) integrate_output(out, newdata=subset(Data,time==t & var=="d1")) )
Est2 = sapply( seq_len(n_t), FUN=\(t) integrate_output(out, newdata=subset(Data,time==t & var=="d2")) )

# True (latent) sample-weighted total
True = tapply( Data$z, INDEX=list("time"=Data$time,"var"=Data$var), FUN=sum )

#
Index = data.frame( expand.grid(dimnames(True)), "True"=as.vector(True) )
Index = data.frame( Index, rbind(t(Est1), t(Est2)) )
Index$low = Index[, 'Est...bias.correct.'] - 1.96*Index[, 'Std..Error']
Index$high = Index[, 'Est...bias.correct.'] + 1.96*Index[, 'Std..Error']

#
library(ggplot2)
ggplot(Index, aes( time, Estimate )) +
  facet_grid( rows=vars(var), scales="free" ) +
  geom_segment(aes(y = low,
                  yend = high,
                  x = time,
                  xend = time) ) +
  geom_point( aes(x=time, y=Estimate), color = "black" ) +
  geom_point( aes(x=time, y=True), color = "red" )
```

