Posterior Sampling and Temporal Prediction Time Analysis

Yifan CHENG y.cheng@u.nus.edu

Simulation Set-Up and Overall Model Fitting Time

When fitting each of our 4 methods (`fullGPfixedL`, `NNGPblockFixedL`, `NNGPsequenFixedL`, and `NNGPsequenVaryLj`) with $m = 8^2 = 64$ and $T = 500$, we specified both `equalTimeDist = TRUE` and `equalTimeDist = FALSE`. We thus have $4 \times 2 = 8$ settings.

Since $m = 64$ is quite small, we do not expect any significant differences in the posterior sampling time between our 4 methods with the same `equalTimeDist` input. Since $T = 500$ is quite large, what we have discussed in Appendix B of our manuscript suggest that the 3 temporal parameters' posterior sampling steps, especially the step for $\boldsymbol{\eta}_t$'s, can be markedly accelerated by adopting our approaches and specifying `equalTimeDist = TRUE` for each of our 4 methods.

The results we have obtained corroborate these well. With `equalTimeDist = TRUE` specified, the overall model fitting time for our 4 methods, i.e., `fullGPfixedL`, `NNGPblockFixedL`, `NNGPsequenFixedL`, and `NNGPsequenVaryLj`, are 7.04 hours, 7.03 hours, 7.04 hours, and 6.31 hours, respectively. If we do not take advantage of our tactics for evenly dispersed time points presented in Appendix B of our manuscript by specifying `equalTimeDist = FALSE` instead, we will need 28.98 days, 29.09 days, 29.03 days, and 28.74 days to fit the same 4 methods using the same computation resources.

Portions of Recorded Gibbs Sampler Time for 10 Key Parameters

We first display the first 50 kept post-burn-in MCMC iterations' posterior sampling time (in milliseconds) for 10 key Gibbs sampler steps (corresponding to $\boldsymbol{\eta}_t$'s, Υ, ψ, $z^o_{jl_j}(\boldsymbol{s}_i)$'s or $u^o_j(\boldsymbol{s}_i)$'s, $\xi^o_j(\boldsymbol{s}_i)$'s, θ_{jl_j}'s, $\delta_{1:k}$, $\alpha^o_{jl_j}(\boldsymbol{s}_i)$'s, κ, and ρ) for our 8 settings.

```r
wd <- paste(projDirec, "simu/appen/equalTimeDistAccelerationVerification/m64T500",
            sep = "/")
setwd(wd)
load("GibbsStepTimeFixedLfullGP.RData")
load("GibbsStepTimeFixedLfullGPfast.RData")
load("GibbsStepTimeFixedLblock.RData")
load("GibbsStepTimeFixedLblockFast.RData")
load("GibbsStepTimeFixedLsequen.RData")
load("GibbsStepTimeFixedLsequenFast.RData")
load("GibbsStepTimeVaryLjSequen.RData")
load("GibbsStepTimeVaryLjSequenFast.RData")
head(GibbsStepTimeFixedLfullGP, 50)      # equalTimeDist = FALSE
```

```
##        z xi theta delta alpha kappa rho   eta upsilon psi
##   [1,] 90 33   296     1     2     1   1 71014       3 128
##   [2,] 85 33   298     1     3     1   1 71013       3 126
##   [3,] 85 33   312     1     3     1   1 71174       3 124
##   [4,] 86 32   297     1     2     1   1 71167       3 182
##   [5,] 84 33   301     1     3     1   1 79008       3 135
##   [6,] 87 31   291     1     3     1   1 79103       3  83
##  [7,] 84 32   305     1     3     1   1 79138       3 180
##  [8,] 88 32   306     1     3     1   1 79408       4  76
##  [9,] 84 32   301     1     2     1   1 79383       3 175
## [10,] 87 32   305     1     3     1   1 79103       3 135
## [11,] 85 31   304     1     2     1   1 75920       4 133
## [12,] 84 32   295     1     3     1   1 75930       4 117
## [13,] 85 33   298     1     3     1   1 75912       4 104
## [14,] 90 32   315     1     3     1   1 76431       4 134
## [15,] 88 32   302     1     3     1   1 76331       3 142
## [16,] 83 32   293     1     3     1   1 76339       3 164
## [17,] 86 32   303     1     2     1   1 69382       4 174
## [18,] 86 32   299     1     3     1   1 75152       4 134
## [19,] 84 32   301     1     3     1   1 75111       4 128
## [20,] 83 33   306     1     3     1   1 75150       4 170
## [21,] 80 32   301     1     3     1   1 72590       4 132
## [22,] 82 32   300     1     3     1   1 72641       4 124
## [23,] 82 32   304     1     3     1   1 72499       4 112
## [24,] 84 32   302     1     3     1   1 72447       4 116
## [25,] 82 32   296     1     3     1   1 72414       3 125
## [26,] 83 32   298     1     3     1   1 74294       4 121
## [27,] 87 32   299     1     3     1   1 73615       4 122
## [28,] 82 34   297     1     3     1   1 73622       4 133
## [29,] 85 33   299     1     3     1   1 65851       4 172
## [30,] 85 32   297     1     3     1   1 74684       4 111
## [31,] 84 32   300     1     3     1   1 74533       4 125
## [32,] 85 33   294     1     3     1   1 80434       3 139
## [33,] 84 32   298     1     3     1   1 67451       3 171
## [34,] 86 32   296     1     3     1   1 74376       3 176
## [35,] 84 32   299     1     3     1   1 77526       3 135
## [36,] 83 32   302     1     3     1   1 77584       3  84
## [37,] 86 33   296     1     3     1   1 77703       4 144
## [38,] 87 33   316     1     3     1   1 77780       3  95
## [39,] 86 33   297     1     3     1   1 77659       3 121
## [40,] 85 33   302     1     3     1   1 77729       3 134
## [41,] 85 33   300     1     3     1   1 77814       3  75
## [42,] 88 33   303     1     3     1   1 77665       3 129
## [43,] 83 33   304     1     3     1   1 76967       3 133
## [44,] 86 33   297     1     3     1   1 76976       3 129
## [45,] 84 33   298     1     3     1   1 76902       3 164
## [46,] 85 32   301     1     3     1   1 69712       3 107
## [47,] 87 33   304     1     3     1   1 69577       3 177
## [48,] 82 32   294     1     3     1   1 77447       4 134
## [49,] 94 31   313     2     3     1   1 77396       4 110
## [50,] 84 32   296     1     3     1   1 79149       3 126
```

```
head(GibbsStepTimeFixedLfullGP.fast, 50) # equalTimeDist = TRUE
```

```
##          z xi theta delta alpha kappa rho eta upsilon psi
##  [1,] 82 32   317     1     3     1   2 327       2  53
##  [2,] 83 34   307     1     3     1   1 330       2  54
##  [3,] 81 32   306     1     3     1   1 322       2  53
##  [4,] 82 33   305     1     3     1   1 323       2  54
##  [5,] 83 34   309     1     3     1   1 324       2  53
##  [6,] 81 33   318     1     3     1   1 329       2  53
##  [7,] 83 33   326     1     3     1   1 329       2  54
##  [8,] 85 32   319     1     3     1   1 327       2  56
##  [9,] 82 33   315     1     3     1   1 325       2  54
## [10,] 84 32   311     1     3     1   2 323       2  54
## [11,] 85 34   323     1     3     1   1 326       2  53
## [12,] 84 34   316     1     3     1   1 323       2  54
## [13,] 87 34   320     1     3     1   1 327       2  53
## [14,] 83 33   321     1     3     1   1 328       2  53
## [15,] 87 32   334     1     3     1   1 327       2  53
## [16,] 84 34   319     1     3     1   2 328       2  53
## [17,] 84 33   328     1     3     1   2 322       2  52
## [18,] 84 32   316     1     3     1   1 319       2  52
## [19,] 82 34   314     1     3     1   1 325       2  53
## [20,] 83 34   311     1     3     1   2 327       2  54
## [21,] 82 33   308     1     3     1   1 324       2  55
## [22,] 82 33   310     1     3     1   2 349       2  55
## [23,] 82 33   333     1     3     1   1 333       2  57
## [24,] 84 34   324     1     3     2   2 325       2  54
## [25,] 82 35   353     1     3     1   2 331       2  53
## [26,] 84 34   324     1     3     1   2 328       2  54
## [27,] 81 33   331     1     3     1   1 325       2  52
## [28,] 83 34   325     1     3     1   2 323       2  52
## [29,] 84 34   318     1     3     1   2 321       2  51
## [30,] 80 34   317     1     3     1   1 323       2  52
## [31,] 82 33   317     1     3     1   2 316       2  54
## [32,] 83 33   317     1     3     1   2 327       2  56
## [33,] 77 33   314     1     3     1   1 326       2  56
## [34,] 83 34   318     1     3     1   2 342       2  57
## [35,] 83 32   323     1     3     1   1 335       2  54
## [36,] 81 33   325     1     2     1   1 330       2  54
## [37,] 81 33   330     1     3     1   2 329       2  55
## [38,] 83 34   327     1     3     1   1 327       2  57
## [39,] 81 32   323     1     3     1   1 323       2  55
## [40,] 83 34   322     1     3     1   1 325       2  55
## [41,] 83 34   319     1     3     1   1 326       2  57
## [42,] 81 34   316     1     3     1   1 329       2  57
## [43,] 84 35   320     1     3     1   1 323       2  56
## [44,] 84 33   321     1     3     1   1 331       2  57
## [45,] 82 32   317     1     3     1   1 338       2  55
## [46,] 84 34   311     1     3     1   1 326       2  53
## [47,] 86 34   319     1     3     1   1 349       2  58
## [48,] 83 32   312     1     3     1   1 338       2  55
## [49,] 82 33   322     1     3     1   1 334       2  56
## [50,] 87 34   342     1     3     1   1 343       2  58
```

```
head(GibbsStepTimeFixedLblock, 50)        # equalTimeDist = FALSE
```

```
##         z xi theta delta alpha kappa rho   eta upsilon psi
##  [1,] 78 33   304     1     3     1   1 70051       3  89
##  [2,] 82 33   300     1     3     1   1 70058       3 175
##  [3,] 80 32   289     1     3     1   1 72111       3 127
##  [4,] 79 31   295     1     3     1   1 72278       3 183
##  [5,] 78 32   294     1     3     1   1 76961       4 110
##  [6,] 79 32   306     1     3     1   1 77133       4 125
##  [7,] 80 32   300     1     3     1   1 77064       4  89
##  [8,] 77 31   296     1     3     1   1 76953       4 139
##  [9,] 78 32   292     1     3     1   1 76967       4 180
## [10,] 78 32   301     1     3     1   1 75239       4 137
## [11,] 76 33   298     1     3     1   1 78195       3 117
## [12,] 79 33   297     1     3     1   1 78087       4 114
## [13,] 78 31   295     1     3     1   1 79100       3 128
## [14,] 78 34   295     1     3     1   1 74809       3  95
## [15,] 78 34   293     1     3     1   1 74774       3 171
## [16,] 78 38   295     1     3     1   1 74843       3 120
## [17,] 78 31   303     1     3     1   1 72534       4 168
## [18,] 77 33   294     1     3     1   1 71906       4 169
## [19,] 75 33   308     1     3     1   1 78745       3  71
## [20,] 76 31   296     1     3     1   1 78715       3 114
## [21,] 74 32   294     1     3     1   1 78714       3 162
## [22,] 75 33   294     1     3     1   1 68979       3 115
## [23,] 76 34   303     1     3     1   1 76470       3 116
## [24,] 78 32   300     1     3     1   1 76493       3 144
## [25,] 74 32   294     1     3     1   1 76353       3 115
## [26,] 75 33   301     1     3     1   1 76295       3 124
## [27,] 75 32   299     1     2     1   1 76245       3 128
## [28,] 76 33   301     1     3     1   1 76313       3 116
## [29,] 76 31   303     1     3     1   1 76298       3 113
## [30,] 76 31   301     1     3     1   1 76257       3 172
## [31,] 74 33   296     1     2     1   1 75355       3 132
## [32,] 72 33   303     1     3     1   1 75327       3 165
## [33,] 72 33   301     1     3     1   1 71289       3 101
## [34,] 72 32   305     1     3     1   1 71270       3 134
## [35,] 72 32   299     1     3     1   1 71438       3 151
## [36,] 72 33   300     1     2     1   1 81806       3 173
## [37,] 74 32   306     1     3     1   1 74861       3  97
## [38,] 79 33   299     1     3     1   1 74938       3 141
## [39,] 81 32   299     1     3     1   1 74865       3 110
## [40,] 79 32   298     1     2     1   1 74812       3 121
## [41,] 78 33   296     1     2     1   1 74797       3 113
## [42,] 78 31   296     1     3     1   1 74818       3 120
## [43,] 76 32   296     1     3     1   1 74819       3 118
## [44,] 74 33   298     1     3     1   1 74936       3 136
## [45,] 75 32   295     1     3     1   1 74759       4 136
## [46,] 76 33   304     1     3     1   1 74887       3 128
## [47,] 75 32   300     1     3     1   1 74891       3 171
## [48,] 78 33   299     1     3     1   1 72076       4 115
## [49,] 77 32   302     1     3     1   1 72072       4 135
## [50,] 77 34   301     1     3     1   1 72056       4 175
```

```
head(GibbsStepTimeFixedLblock.fast, 50)  # equalTimeDist = TRUE
```

```
##          z xi theta delta alpha kappa rho eta upsilon psi
##  [1,]  82 35   325     1     3     1   1 324       2  52
##  [2,]  85 34   322     1     3     1   1 324       2  51
##  [3,]  82 33   314     1     3     1   1 313       2  51
##  [4,]  83 35   317     1     3     1   2 315       2  52
##  [5,]  81 32   318     1     3     1   1 323       2  52
##  [6,]  86 34   313     1     3     1   1 330       2  54
##  [7,]  84 35   307     1     3     1   1 327       2  53
##  [8,]  81 33   312     1     3     1   1 329       2  54
##  [9,]  83 34   314     1     3     1   1 325       2  55
## [10,]  84 32   315     1     3     1   1 326       2  54
## [11,]  81 35   321     1     3     1   1 324       2  54
## [12,]  84 32   322     1     3     1   1 330       2  53
## [13,]  83 32   315     1     3     1   1 324       2  54
## [14,]  85 33   319     1     3     1   1 321       2  53
## [15,]  85 31   315     1     3     1   1 321       2  52
## [16,]  82 33   312     1     3     1   1 316       2  53
## [17,]  82 32   309     1     3     1   1 326       2  52
## [18,]  82 32   317     1     3     1   1 328       2  52
## [19,]  83 32   324     1     3     1   1 322       2  52
## [20,]  84 33   317     1     3     1   1 330       2  53
## [21,]  84 34   330     1     3     1   1 332       2  55
## [22,]  85 32   314     1     3     1   1 329       2  54
## [23,]  84 33   319     1     3     1   1 324       2  54
## [24,]  87 32   310     1     3     1   1 334       2  54
## [25,]  86 33   310     1     3     1   1 335       2  56
## [26,]  83 33   315     1     3     1   1 336       2  53
## [27,]  85 33   321     1     3     1   1 347       2  55
## [28,]  84 33   322     1     3     1   1 328       2  53
## [29,]  83 32   324     1     3     1   1 329       2  53
## [30,]  90 34   321     1     3     1   1 325       2  60
## [31,]  89 33   340     1     3     1   1 325       2  52
## [32,]  82 33   316     1     3     1   1 314       2  52
## [33,]  82 33   317     1     3     1   1 325       2  51
## [34,]  85 34   313     1     3     1   1 334       2  53
## [35,]  81 33   314     1     3     1   1 339       2  55
## [36,]  81 32   306     1     3     1   1 333       2  55
## [37,]  81 32   315     1     3     1   1 330       2  54
## [38,]  82 32   314     1     3     1   1 328       2  54
## [39,]  82 32   311     1     3     1   1 328       2  56
## [40,]  80 33   318     1     3     1   1 326       2  54
## [41,]  82 32   317     1     3     1   1 327       2  55
## [42,]  87 32   315     1     3     1   1 320       2  53
## [43,]  80 33   318     1     3     1   1 321       2  53
## [44,]  82 33   319     1     3     1   1 325       2  55
## [45,]  82 33   318     1     3     1   1 325       2  57
## [46,]  82 32   318     1     3     1   1 329       2  55
## [47,]  82 34   318     1     3     1   1 324       2  54
## [48,] 218 34   318     1     3     1   1 318       2  53
## [49,]  81 34   318     1     3     1   1 321       2  55
## [50,]  85 33   313     1     3     1   1 317       2  54
```

```
head(GibbsStepTimeFixedLsequen, 50)       # equalTimeDist = FALSE
```

```
##         z xi theta delta alpha kappa rho   eta upsilon psi
##   [1,] 79 33   296     1     9     1   1 75497       4 123
##   [2,] 82 33   295     1     9     1   1 75417       4  82
##   [3,] 81 32   298     1     9     1   1 75276       4 133
##   [4,] 80 32   300     1     9     1   1 75807       4 143
##   [5,] 80 33   301     1     9     1   1 79764       4 142
##   [6,] 79 34   310     1     9     1   1 75835       4  89
##   [7,] 80 33   317     1     9     1   1 75846       3 125
##   [8,] 82 34   308     1     9     1   1 75104       4 180
##   [9,] 81 33   309     1     9     1   1 79590       3  93
##  [10,] 81 31   305     1     9     1   1 79517       3 175
## [11,] 79 33   305     1     9     1   1 75497       4 101
## [12,] 80 33   304     1     9     1   1 75502       4 138
## [13,] 79 34   305     1     9     1   1 75970       4 123
## [14,] 81 34   305     1     9     1   1 75928       4 123
## [15,] 82 33   306     1     9     1   1 75637       4 137
## [16,] 82 33   313     1     9     1   1 75608       4 140
## [17,] 84 32   302     1     9     1   2 75773       4 181
## [18,] 81 32   304     1     9     1   1 76129       4 129
## [19,] 80 32   308     1     9     1   1 76219       4 139
## [20,] 82 32   310     1     9     1   1 76602       4 182
## [21,] 81 33   307     1     9     1   1 77476       4 100
## [22,] 81 32   303     1     9     1   1 73108       4 146
## [23,] 79 33   304     1     9     1   1 73819       4 137
## [24,] 83 35   308     1     9     1   1 73762       3 101
## [25,] 83 31   303     1     9     1   1 73533       3  74
## [26,] 83 33   299     1     9     1   1 73859       4 134
## [27,] 82 34   302     1     9     1   1 74083       3 172
## [28,] 78 34   304     1     9     1   1 69390       4 177
## [29,] 81 33   303     1     9     1   1 74331       4 111
## [30,] 81 32   304     1     9     1   1 80581       3 141
## [31,] 84 34   310     1     9     1   1 80485       4 136
## [32,] 81 32   308     1     9     1   1 72570       4 129
## [33,] 81 34   302     1     9     1   1 72510       4 110
## [34,] 75 33   311     1     9     1   1 73467       4 101
## [35,] 76 34   304     1     9     1   1 77116       3 179
## [36,] 79 32   302     1     9     1   1 82984       4 141
## [37,] 80 34   310     1     9     1   1 76213       4 137
## [38,] 79 32   300     1     9     1   1 76644       3  92
## [39,] 75 32   300     1     9     1   1 76660       3 164
## [40,] 78 32   298     1     9     1   1 79264       4 101
## [41,] 79 32   301     1     9     1   1 78980       4  89
## [42,] 81 33   308     1     9     1   1 79446       4 174
## [43,] 85 32   317     1     9     1   1 75268       4 117
## [44,] 79 31   309     1     9     1   1 74430       4  78
## [45,] 86 33   314     1     9     1   1 73990       4 145
## [46,] 78 32   306     1     9     1   1 73787       4 117
## [47,] 78 32   300     1     9     1   1 74215       4 134
## [48,] 85 33   305     1     9     1   1 76137       3 126
## [49,] 82 33   309     1     9     1   1 76173       4 138
## [50,] 83 32   303     1     9     1   1 75553       3 179
```

```
head(GibbsStepTimeFixedLsequen.fast, 50) # equalTimeDist = TRUE
```

```
##        z xi theta delta alpha kappa rho eta upsilon psi
##  [1,] 75 33   321     1     9     1   1 327       2  57
##  [2,] 77 34   318     1     9     1   1 325       2  57
##  [3,] 80 33   322     1     9     1   1 329       2  56
##  [4,] 73 33   316     1    12     4   3 336       6  60
##  [5,] 77 32   316     1     9     1   1 331       2  56
##  [6,] 77 34   319     1     9     2   1 328       2  53
##  [7,] 76 32   327     1     9     1   1 330       2  54
##  [8,] 76 33   322     1     9     1   1 324       2  53
##  [9,] 79 34   321     1     9     1   1 326       2  53
## [10,] 79 33   328     1     9     1   1 328       2  54
## [11,] 74 32   319     1     9     1   1 323       2  54
## [12,] 76 34   320     1     9     1   1 326       2  53
## [13,] 76 34   323     1     9     1   1 323       2  53
## [14,] 78 34   327     1     9     1   1 323       2  53
## [15,] 79 35   319     1     9     1   2 321       2  53
## [16,] 78 32   320     1     9     1   1 339       2  53
## [17,] 76 32   318     1     9     1   1 332       2  55
## [18,] 78 34   315     1     9     1   2 332       2  54
## [19,] 75 33   311     1     9     1   2 328       2  53
## [20,] 75 32   323     1     9     1   1 330       2  53
## [21,] 77 32   329     1     9     1   2 341       2  54
## [22,] 75 32   329     1     9     1   2 329       2  53
## [23,] 81 33   322     1     9     1   2 325       2  54
## [24,] 75 32   321     1     9     1   1 322       2  56
## [25,] 77 34   323     1     9     1   1 324       2  53
## [26,] 76 34   324     1     9     1   1 325       2  54
## [27,] 76 33   320     1     9     1   2 327       2  53
## [28,] 78 33   322     1     9     1   1 318       2  54
## [29,] 76 32   309     1     9     1   1 335       2  55
## [30,] 75 32   311     1     9     1   2 326       2  53
## [31,] 77 34   316     1     9     1   2 331       2  55
## [32,] 75 33   316     1     9     1   1 331       2  55
## [33,] 78 33   320     1     9     1   1 332       2  55
## [34,] 76 34   330     1     9     1   2 334       2  56
## [35,] 76 33   320     1     9     1   1 325       2  55
## [36,] 78 33   321     1     9     1   1 326       2  57
## [37,] 78 34   326     1     9     1   2 331       2  57
## [38,] 78 34   322     1     9     1   1 333       2  57
## [39,] 78 36   332     1     9     1   1 330       2  55
## [40,] 76 33   317     1     9     1   2 323       2  56
## [41,] 82 33   324     1     9     1   1 320       2  55
## [42,] 78 35   323     1     9     1   2 322       2  56
## [43,] 76 32   314     1     9     1   2 328       2  54
## [44,] 80 34   318     1     9     1   1 330       2  53
## [45,] 76 32   313     1     9     1   1 331       2  53
## [46,] 80 35   317     1     9     1   2 330       2  54
## [47,] 77 33   314     1     9     1   1 327       2  54
## [48,] 77 34   320     1     9     1   2 332       2  53
## [49,] 79 33   317     1     9     1   1 333       2  54
## [50,] 76 32   320     1     9     1   2 333       2  53
```

```
head(GibbsStepTimeVaryLjSequen, 50)      # equalTimeDist = FALSE
```

```
##       u xi theta delta alpha kappa rho   eta upsilon psi
##  [1,] 0  6   293     1    27     1   1 79983       4 114
##  [2,] 0  6   311     1    27     1   1 80032       3 161
##  [3,] 0  6   295     1    27     0   1 67313       4 112
##  [4,] 1  6   301     1    28     1   1 67432       4 122
##  [5,] 0  6   296     1    27     1   1 70965       3 113
##  [6,] 0  6   299     1    27     1   1 75507       3 106
##  [7,] 0  6   299     1    28     1   1 70782       3 130
##  [8,] 0  6   304     1    27     1   1 72513       4  97
##  [9,] 1  6   307     1    28     1   1 66166       3  93
## [10,] 0  6   293     1    27     1   1 72674       3  90
## [11,] 0  6   293     1    27     1   1 72849       4 134
## [12,] 0  6   293     1    26     1   1 71106       3 118
## [13,] 0  6   297     1    27     1   1 75151       4 124
## [14,] 0  6   298     1    28     1   1 75084       4 172
## [15,] 0  6   306     1    28     1   1 74628       3  99
## [16,] 1  6   293     1    28     1   1 74831       4  92
## [17,] 0  6   295     1    31     1   1 74632       4 120
## [18,] 0  6   305     1    30     1   1 74668       3  76
## [19,] 0  6   299     1    30     1   1 74550       3 116
## [20,] 0  6   294     1    29     1   1 74524       3 110
## [21,] 0  6   298     1    30     1   1 74559       3  73
## [22,] 0  6   295     1    30     1   1 78794       3 121
## [23,] 0  6   294     1    30     1   1 75168       4 108
## [24,] 0  6   299     1    30     1   1 75156       4  90
## [25,] 0  6   295     1    30     0   1 75537       4  96
## [26,] 0  6   294     1    30     1   1 75647       4 121
## [27,] 0  6   289     1    29     1   1 75524       4  89
## [28,] 1  6   301     1    30     1   1 75675       4 126
## [29,] 1  6   307     1    30     0   1 77502       4 126
## [30,] 0  6   296     1    30     1   1 73704       4 125
## [31,] 0  6   302     1    30     1   1 73637       3 105
## [32,] 0  6   301     1   153     1   1 73663       4 164
## [33,] 0  6   298     1    28     1   1 69510       4 107
## [34,] 0  6   302     1    28     1   1 69565       4  81
## [35,] 0  6   301     1    28     1   1 69528       4 142
## [36,] 0  6   292     1    28     1   1 69894       4 142
## [37,] 0  6   289     1    28     0   1 70063       4 181
## [38,] 0  6   293     1    28     1   1 79009       4 131
## [39,] 0  6   292     1    28     1   1 79050       3  95
## [40,] 0  6   303     1    28     1   1 78990       3 117
## [41,] 0  6   301     1    28     1   1 79005       4  97
## [42,] 0  6   303     1    28     1   1 79039       4 178
## [43,] 0  6   298     1    28     1   1 78449       4 129
## [44,] 0  6   297     1    28     0   1 78405       4 139
## [45,] 0  6   290     1    28     1   1 78462       4  93
## [46,] 1  6   299     1    28     1   1 80548       3 179
## [47,] 0  6   295     1    28     1   1 79434       4 108
## [48,] 0  6   300     1    28     1   1 79661       4 125
## [49,] 0  6   291     1    28     1   1 75021       4 140
## [50,] 0  6   299     1    27     1   1 75002       3 112
```

```r
head(GibbsStepTimeVaryLjSequen.fast, 50) # equalTimeDist = TRUE
```

```
##       u xi theta delta alpha kappa rho eta upsilon psi
##  [1,] 1  7   310     1    33     1   1 324       2  54
##  [2,] 1  7   320     1    32     1   1 331       2  54
##  [3,] 1  7   314     1    30     1   1 326       2  54
##  [4,] 1  7   317     1    32     1   1 327       2  53
##  [5,] 1  8   319     1    33     1   1 324       2  54
##  [6,] 1  7   317     1    40     1   1 322       2  52
##  [7,] 1  7   313     1    30     1   1 328       2  55
##  [8,] 1  7   319     1    32     1   1 327       2  54
##  [9,] 1  7   312     1    31     1   1 323       2  55
## [10,] 1  7   317     1    31     1   1 329       2  56
## [11,] 1  7   309     1    32     1   1 325       2  55
## [12,] 1  7   313     1    33     1   1 321       2  54
## [13,] 1  7   312     1    35     1   1 318       2  53
## [14,] 1  7   303     1    32     1   1 327       2  53
## [15,] 1  7   304     1    33     1   1 328       2  53
## [16,] 1  7   310     1    31     1   1 321       2  56
## [17,] 1  7   308     1    31     1   1 325       2  56
## [18,] 0  6   312     1    32     1   1 321       2  55
## [19,] 1  7   313     1    33     1   1 321       2  54
## [20,] 1  7   314     1    34     1   1 318       2  53
## [21,] 1  7   317     1    33     1   1 319       2  52
## [22,] 1  7   311     1    32     1   1 312       2  52
## [23,] 1  7   313     1    32     1   1 313       2  52
## [24,] 1  7   311     1    32     1   1 312       2  53
## [25,] 1  7   316     1    30     1   1 321       2  54
## [26,] 1  7   320     1    33     1   1 316       2  54
## [27,] 1  7   312     1    34     1   1 315       2  52
## [28,] 1  7   321     1    34     1   1 324       2  53
## [29,] 1  7   324     1    33     1   1 321       2  54
## [30,] 1  7   322     1    33     1   1 323       2  53
## [31,] 1  7   319     1    31     1   1 319       2  57
## [32,] 1  7   321     1    31     1   1 324       2  54
## [33,] 1  7   318     1    32     1   1 327       2  55
## [34,] 1  7   320     1    33     1   1 335       2  55
## [35,] 1  7   317     1    35     1   1 323       2  53
## [36,] 1  7   321     1    35     1   1 321       2  53
## [37,] 1  7   328     1    33     1   1 328       2  53
## [38,] 1  7   327     1    33     1   1 325       2  55
## [39,] 1  6   325     1    32     1   1 324       2  55
## [40,] 1  6   321     1    31     1   1 325       2  54
## [41,] 1  7   315     1    32     1   1 325       2  53
## [42,] 1  6   309     1    33     0   1 324       2  52
## [43,] 1  7   314     1    35     1   1 327       2  54
## [44,] 1  7   315     1    32     1   1 326       2  54
## [45,] 1  7   322     1    33     1   1 321       2  54
## [46,] 1  7   314     1    32     1   1 325       2  55
## [47,] 1  7   319     1    31     1   1 324       2  55
## [48,] 1  7   318     1    30     1   1 321       2  53
## [49,] 1  6   315     1    33     1   1 324       2  52
## [50,] 1  7   314     1    34     1   1 328       2  53
```

Posterior Sampling Time Summary Statistics for the 3 Temporal Parameters

We then present vital posterior sampling time summary statistics for the 3 temporal parameters $\boldsymbol{\eta}_t$'s, Υ, and ψ to showcase the manifest computational acceleration brought about by our tactics for evenly dispersed time points presented in Appendix B.

```
fullGPfixedL.slow <- apply(GibbsStepTimeFixedLfullGP[,8:10], 2, summary)
fullGPfixedL.fast <- apply(GibbsStepTimeFixedLfullGP.fast[,8:10], 2, summary)
NNGPblockFixedL.slow <- apply(GibbsStepTimeFixedLblock[,8:10], 2, summary)
NNGPblockFixedL.fast <- apply(GibbsStepTimeFixedLblock.fast[,8:10], 2, summary)
NNGPsequenFixedL.slow <- apply(GibbsStepTimeFixedLsequen[,8:10], 2, summary)
NNGPsequenFixedL.fast <- apply(GibbsStepTimeFixedLsequen.fast[,8:10], 2, summary)
NNGPsequenVaryLj.slow <- apply(GibbsStepTimeVaryLjSequen[,8:10], 2, summary)
NNGPsequenVaryLj.fast <- apply(GibbsStepTimeVaryLjSequen.fast[,8:10], 2, summary)
```

```
fullGPfixedLsummary <- data.frame(eta = fullGPfixedL.slow[,1],
                                  eta.fast = fullGPfixedL.fast[,1],
                                  upsilon = fullGPfixedL.slow[,2],
                                  upsilon.fast = fullGPfixedL.fast[,2],
                                  psi = fullGPfixedL.slow[,3],
                                  psi.fast = fullGPfixedL.fast[,3])
NNGPblockFixedLsummary <- data.frame(eta = NNGPblockFixedL.slow[,1],
                                     eta.fast = NNGPblockFixedL.fast[,1],
                                     upsilon = NNGPblockFixedL.slow[,2],
                                     upsilon.fast = NNGPblockFixedL.fast[,2],
                                     psi = NNGPblockFixedL.slow[,3],
                                     psi.fast = NNGPblockFixedL.fast[,3])
NNGPsequenFixedLsummary <- data.frame(eta = NNGPsequenFixedL.slow[,1],
                                      eta.fast = NNGPsequenFixedL.fast[,1],
                                      upsilon = NNGPsequenFixedL.slow[,2],
                                      upsilon.fast = NNGPsequenFixedL.fast[,2],
                                      psi = NNGPsequenFixedL.slow[,3],
                                      psi.fast = NNGPsequenFixedL.fast[,3])
NNGPsequenVaryLjsummary <- data.frame(eta = NNGPsequenVaryLj.slow[,1],
                                      eta.fast = NNGPsequenVaryLj.fast[,1],
                                      upsilon = NNGPsequenVaryLj.slow[,2],
                                      upsilon.fast = NNGPsequenVaryLj.fast[,2],
                                      psi = NNGPsequenVaryLj.slow[,3],
                                      psi.fast = NNGPsequenVaryLj.fast[,3])
fullGPfixedLsummary
```

```
##              eta eta.fast upsilon upsilon.fast      psi psi.fast
## Min.    47910.00 307.0000  3.0000       2.0000  68.0000  51.0000
## 1st Qu. 71600.25 326.0000  3.0000       2.0000 114.0000  54.0000
## Median  74556.00 333.0000  3.0000       2.0000 129.0000  55.0000
## Mean    73937.04 334.2716  3.4484       2.0234 131.1978  54.9572
## 3rd Qu. 77246.50 342.0000  4.0000       2.0000 152.2500  56.0000
## Max.    81935.00 403.0000  5.0000       6.0000 192.0000  73.0000
```

```
NNGPblockFixedLsummary
```

```
##              eta eta.fast upsilon upsilon.fast      psi psi.fast
## Min.    49929.00 309.0000  3.0000       2.0000  69.0000  51.0000
## 1st Qu. 72014.75 326.0000  3.0000       2.0000 114.0000  54.0000
## Median  75084.00 333.0000  3.0000       2.0000 129.0000  55.0000
## Mean    74305.95 334.8152  3.4336       2.0162 131.7338  55.0994
## 3rd Qu. 77423.25 342.0000  4.0000       2.0000 157.0000  56.0000
## Max.    81945.00 417.0000  5.0000       5.0000 193.0000  72.0000
```

```
NNGPsequenFixedLsummary
```

```
##               eta eta.fast upsilon upsilon.fast      psi psi.fast
## Min.     51226.00 307.0000  3.0000       2.0000  69.0000  50.0000
## 1st Qu.  72183.25 326.0000  3.0000       2.0000 116.0000  54.0000
## Median   75226.00 333.0000  4.0000       2.0000 131.0000  55.0000
## Mean     76910.86 335.0714  3.5734       2.0292 136.7524  55.1294
## 3rd Qu.  78106.50 342.0000  4.0000       2.0000 164.0000  56.0000
## Max.    129933.00 415.0000  6.0000       6.0000 285.0000  72.0000
```

```
NNGPsequenVaryLjsummary
```

```
##                eta eta.fast upsilon upsilon.fast      psi psi.fast
## Min.     42653.00 306.0000  3.0000       2.0000  68.0000   51.000
## 1st Qu.  70463.75 323.0000  3.0000       2.0000 113.0000   54.000
## Median   74632.50 328.0000  4.0000       2.0000 129.0000   54.000
## Mean     75795.19 330.1242  3.5962       2.0008 133.8518   54.725
## 3rd Qu.  77780.25 335.0000  4.0000       2.0000 159.0000   56.000
## Max.    134470.00 390.0000  6.0000       4.0000 295.0000   69.000
```

The results correspond well to what we have discussed in Appendix B of our manuscript. **For each of our 4 methods, specifying `equalTimeDist = TRUE` instead of `equalTimeDist = FALSE` on equispaced time points markedly accelerates posterior Gibbs sampler steps of $\boldsymbol{\eta}_t$'s, ψ, and Υ (especially the step for $\boldsymbol{\eta}_t$'s).**

We finally calculate standard deviations for the 3 temporal-related parameters' posterior sampling time across all kept post-burn-in MCMC iterations.

```
round(apply(GibbsStepTimeFixedLfullGP[,8:10], 2, sd), 5)
```

```
##        eta    upsilon        psi
## 4305.40218    0.49859   28.33158
```

```
round(apply(GibbsStepTimeFixedLfullGP.fast[,8:10], 2, sd), 5)
```

```
##      eta  upsilon      psi
## 11.53834  0.22375  1.98357
```

```
round(apply(GibbsStepTimeFixedLblock[,8:10], 2, sd), 5)
```

```
##        eta    upsilon        psi
## 4412.38304    0.49602   28.56903
```

```
round(apply(GibbsStepTimeFixedLblock.fast[,8:10], 2, sd), 5)
```

```
##      eta  upsilon      psi
## 11.68779  0.15473  1.96598
```

```
round(apply(GibbsStepTimeFixedLsequen[,8:10], 2, sd), 5)
```

```
##         eta     upsilon         psi
## 10030.28478     0.62824    33.26132
```

```
round(apply(GibbsStepTimeFixedLsequen.fast[,8:10], 2, sd), 5)
```

```
##      eta  upsilon      psi
## 12.17556  0.25992  2.00416
```

```
round(apply(GibbsStepTimeVaryLjSequen[,8:10], 2, sd), 5)
```

```
##         eta     upsilon         psi
## 10332.05841     0.63626    34.01907
```

```
round(apply(GibbsStepTimeVaryLjSequen.fast[,8:10], 2, sd), 5)
```

```
##      eta  upsilon      psi
## 10.41849  0.03464  1.77392
```

Time Required for Predicting at Future Time Points

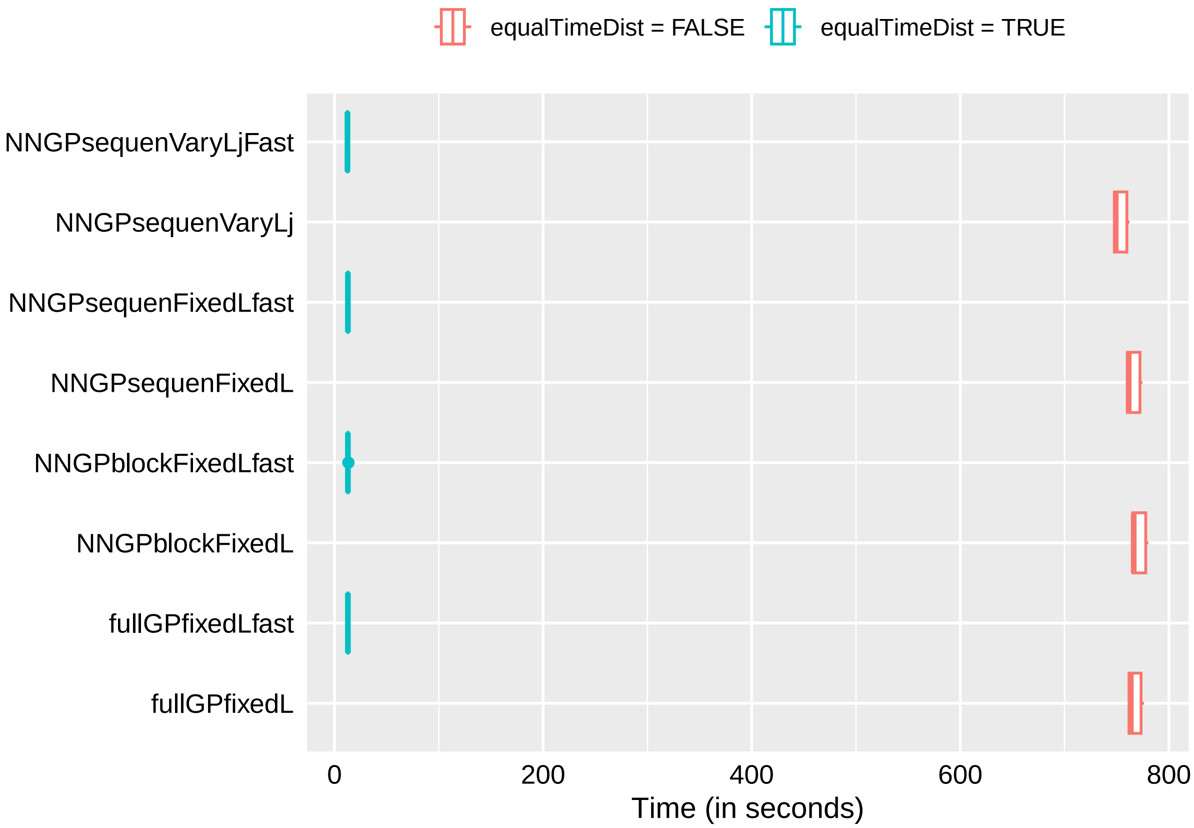
For each of our 8 settings, we record time needed to predict at 10 future time points. We obtain 100 temporal prediction instances based on each of our 8 obtained model fitting objects and thus have 100 × 8 recorded time intervals for analysis.

```r
rm(list=ls())
library(spatempBFA)
modelVec <- c("fullGPfixedLfast", "fullGPfixedL", "NNGPblockFixedLfast", "NNGPblockFixedL",
              "NNGPsequenFixedLfast", "NNGPsequenFixedL",
              "NNGPsequenVaryLjFast", "NNGPsequenVaryLj")
N <- 100
Nu <- 500
newT <- 10
temppredTimeMat <- matrix(0, N, 8)
colnames(temppredTimeMat) <- modelVec
load("regFixedL30simuT500M64Iter30000.RData")
regFixedL.simu.fast <- regFixedL.simu
load("regFixedL30simuT500M64Iter30000specifyEqualTimeDistF.RData")
load("regFixedL30simuBlockT500M64Iter30000.RData")
regFixedL.simu.block.fast <- regFixedL.simu.block
load("regFixedL30simuBlockT500M64Iter30000specifyEqualTimeDistF.RData")
load("regFixedL30simuSequenT500M64Iter30000nostorealphaweights.RData")
regFixedL.simu.sequen.fast <- regFixedL.simu.sequen
load("regFixedL30simuSequenT500M64Iter30000specifyEqualTimeDistFnostorealphaweights.RData")
load("regVaryLjsimuSequenT500M64Iter30000nostorealphaweight.RData")
regVaryLj.simu.sequen.fast <- regVaryLj.simu.sequen
load("regVaryLjsimuSequenT500M64Iter30000specifyEqualTimeDistFnostorealphaweight.RData")
for (n in 1:N) {
  print(n)
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu.fast, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 1] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 2] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu.block.fast, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 3] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu.block, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 4] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu.sequen.fast, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 5] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regFixedL.simu.sequen, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 6] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regVaryLj.simu.sequen.fast, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 7] = difftime(t2, t1, units = "secs")
  t1 <- Sys.time()
  temppredobj <- predictNewTime(regVaryLj.simu.sequen, (Nu+1):(Nu+newT), seed = 29)
  t2 <- Sys.time()
  temppredTimeMat[n, 8] = difftime(t2, t1, units = "secs")
}
save(temppredTimeMat, file = "m64T500temppredTimeMat.RData")
```

```
setwd(wd)
load("m64T500temppredTimeMat.RData")
apply(temppredTimeMat, 2, summary)
```

```
##         fullGPfixedLfast fullGPfixedL NNGPblockFixedLfast NNGPblockFixedL
## Min.            12.50355     760.7941            12.56436        764.6271
## 1st Qu.         12.72893     761.7930            12.77132        765.5061
## Median          12.85846     763.4865            12.87959        766.2839
## Mean            12.85741     767.6316            12.88393        771.5122
## 3rd Qu.         12.96917     773.3529            12.99655        777.8104
## Max.            13.31018     776.0580            13.34953        780.2927
##         NNGPsequenFixedLfast NNGPsequenFixedL NNGPsequenVaryLjFast
## Min.                12.48124         758.8342             12.17900
## 1st Qu.             12.75420         760.2494             12.39718
## Median              12.86878         761.4499             12.46448
## Mean                12.88437         766.1333             12.49416
## 3rd Qu.             13.02783         772.2879             12.58538
## Max.                13.34398         774.6204             12.82533
##         NNGPsequenVaryLj
## Min.            746.8558
## 1st Qu.         747.8390
## Median          749.0556
## Mean            753.7606
## 3rd Qu.         759.7292
## Max.            762.1863
```

```
library(tidyverse)
library(ggpubr)
N <- nrow(temppredTimeMat)
equalTimeDistTF = as.factor(rep(rep(c("equalTimeDist = TRUE",
                                      "equalTimeDist = FALSE"), each = N), 4))
temppredTimeDF <- data.frame(temppredTime = as.vector(temppredTimeMat),
                             model = as.factor(rep(colnames(temppredTimeMat), each = N)),
                             equalTimeDistTF = equalTimeDistTF)
temppredtimeBox <- ggplot(temppredTimeDF) + labs(y = "", x = "Time (in seconds)") +
  geom_boxplot(aes(x = temppredTime, y = model, color = equalTimeDistTF)) +
  #scale_fill_manual("", values = c("#55CC11", "#1177CC")) +
  theme(axis.text.x = element_text(size = 10, color = "black"),
        axis.text.y = element_text(size = 10, color = "black"),
        axis.ticks.x = element_blank(), axis.ticks.y = element_blank(),
        legend.position = "top", legend.key = element_blank(),
        legend.title = element_blank())
temppredtimeBox
```



```
# ggsave("tempPredBox.png", width = 16, height = 10, units = "cm")
```

For each of our 4 methods, specifying `equalTimeDist = TRUE` instead of `equalTimeDist = FALSE` on equispaced time points markedly accelerates the step obtaining $\hat{\boldsymbol{\eta}}_{(T+1):(T+q)}$ given $\boldsymbol{\eta}_{1:T}$, Υ, ψ when making future-time predictions, as we no longer need to evaluate $H(\psi)^{-1}$ in each MCMC iteration. The overall temporal prediction time is thus considerably reduced, as expected. See Appendices B.1, B.3.3, and G.1 for more details.

Data Generation and Model Fitting

```
rm(list=ls())
library(mvtnorm)
library(fields)
library(spatempBFA)
library(coda)
K <- 5
O <- 1
L <- 30
M <- 64
LjVec <- rep(min(30, M), K)
sqrootM <- 8
Nu <- 500
Time <- 1:Nu
TimeDist <- as.matrix(dist(Time))
APsi = 0.1; BPsi = 4.5
```

```
set.seed(29)
### 1) actual sigma^2(i,o) (for i=1,2,...,M and o=1) values
sigma2 <- 0.01
### 2) actual psi value
psi <- 2.3
### 3) actual kappa value
kappa <- 0.7
### 4) actual Upsilon(K\times K)
tempMat <- matrix(runif(K*K,0,1),K,K)
Upsilon <- t(tempMat)%*%tempMat
### 5) actual rho value
rho <- 0.8
D <- rdist(expand.grid(1:sqrootM, 1:sqrootM))
Frho <- exp(-rho*D)
### 6) actual Eta (c(Eta_1,...,Eta_T)) (vec of length Nu*K)
Hpsi <- exp(-psi*TimeDist)
Eta <- rmvnorm(1, mean=rep(0, Nu*K), sigma=kronecker(Hpsi, Upsilon))
## Y~0 (P=0) so no need to sample Beta; all familyInd=0 (normal) so no need to sample Y
maxL <- 10
LStarJ <- sample(maxL, size=K, replace=T)
### 7) actual alpha
Alpha <- list()
for(j in 1:K) {
  Alpha[[j]] <-  t(rmvnorm(LStarJ[j], mean=rep(0,M*O), sigma=kappa*Frho))
  #every list index an M by L_j matrix
}
w <- list()
for(j in 1:K){
  w[[j]] <- pnorm(Alpha[[j]])
  Lj <- LStarJ[j]
  w[[j]][,Lj] <- rep(1, M)
  temp <- rep(1, M)
  for(l in 1:Lj){
    w[[j]][,l] <- w[[j]][,l]*temp
    if(l<Lj) {temp <- temp * pnorm(Alpha[[j]][,l], lower.tail = FALSE)}
  }
}
### 8) actual Xi
Xi <- matrix(1, M, K)
for(j in 1:K){
  Lj <- LStarJ[j]
  for(i in 1:M){
    Xi[i,j] <- sample(Lj, size=1, prob=w[[j]][i,])
  }
}
### 9) actual Delta
a1=1; a2=10
Delta <- sapply(c(a1,rep(a2,(K-1))), rgamma, n=1, rate=1) # Tau <- cumprod(Delta)
### 10) actual Theta
Theta <- list()
for(j in 1:K){
  Theta[[j]] <- rnorm(LStarJ[j], 0, sd=sqrt(1/Delta[j])) #vector of length Lj
}
```

```
Lambda <- matrix(0, M*O, K)
for(j in 1:K){
  for(i in 1:M){
    Lambda[i,j] = Theta[[j]][Xi[i,j]]
  }
}
Sigma.NuMO <- rnorm(Nu * M * O, sd = sqrt(sigma2))
EtaMat <- matrix(Eta, K, Nu)
meanMat <- Lambda%*%EtaMat #M*O\times Nu
Yobs <- as.vector(meanMat) + Sigma.NuMO
dat <- data.frame(Y = Yobs)
Hypers <- list(Sigma2 = list(A = 1, B = 1), Rho = list(ARho=0.1, BRho=1),
               Kappa = list(SmallUpsilon = O + 1, BigTheta = diag(O)),
               Psi = list(APsi = APsi, BPsi = BPsi),
               Upsilon = list(Zeta = K + 1, Omega = diag(K)))
MCMC <- list(NBurn = 20000, NSims = 10000, NThin = 2, NPilot = 5)
regFixedL.simu <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                            starting = NULL, hypers = Hypers, tuning = NULL,
                            mcmc = MCMC,
                            L = L,
                            family = "normal",
                            temporal.structure = "exponential",
                            spatial.structure = "continuous",
                            seed = 29,
                            gamma.shrinkage = TRUE,
                            include.time = TRUE,
                            include.space = TRUE,
                            clustering = TRUE,
                            seasonPeriod = 1,
                            equalTimeDist = TRUE,
                            spatApprox = FALSE,
                            alphaMethod = "block",
                            h = 15,
                            storeSpatPredPara = FALSE,
                            storeWeights = FALSE,
                            alphasWeightsToFiles = FALSE)
save(regFixedL.simu, file="regFixedL30simuT500M64Iter30000.RData")
Diags <- diagnostics(regFixedL.simu, diags = c("dic", "dinf", "meanIC", "waic"),
                     keepDeviance = TRUE)
save(Diags, file = "regFixedL30simuT500M64Iter30000Diags.RData")
Deviance <- as.mcmc(Diags$deviance)
save(Deviance, file = "regFixedL30simuT500M64Iter30000Deviance.RData")
GibbsStepTimeFixedLfullGP.fast <- regFixedL.simu$GibbsStepTime
save(GibbsStepTimeFixedLfullGP.fast, file = "GibbsStepTimeFixedLfullGPfast.RData")
regFixedL.simu <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                            starting = NULL, hypers = Hypers, tuning = NULL,
                            mcmc = MCMC,
                            L = L,
                            family = "normal",
                            temporal.structure = "exponential",
                            spatial.structure = "continuous",
                            seed = 29,
                            gamma.shrinkage = TRUE,
                            include.time = TRUE,
                            include.space = TRUE,
                            clustering = TRUE,
                            seasonPeriod = 1,
                            equalTimeDist = FALSE,
                            spatApprox = FALSE,
                            alphaMethod = "block",
                            h = 15,
                            storeSpatPredPara = FALSE,
                            storeWeights = FALSE,
                            alphasWeightsToFiles = FALSE)
save(regFixedL.simu,
     file="regFixedL30simuT500M64Iter30000specifyEqualTimeDistF.RData")
Diags <- diagnostics(regFixedL.simu, diags = c("dic", "dinf", "meanIC", "waic"),
                     keepDeviance = TRUE)
save(Diags,
     file = "regFixedL30simuT500M64Iter30000specifyEqualTimeDistFDiags.RData")
Deviance <- as.mcmc(Diags$deviance)
save(Deviance,
     file = "regFixedL30simuT500M64Iter30000specifyEqualTimeDistFDeviance.RData")
GibbsStepTimeFixedLfullGP <- regFixedL.simu$GibbsStepTime
save(GibbsStepTimeFixedLfullGP, file = "GibbsStepTimeFixedLfullGP.RData")
regFixedL.simu.block <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                  starting = NULL, hypers = Hypers, tuning = NULL,
                                  mcmc = MCMC,
                                  L = L,
                                  family = "normal",
                                  temporal.structure = "exponential",
                                  spatial.structure = "continuous",
                                  seed = 29,
                                  gamma.shrinkage = TRUE,
                                  include.time = TRUE,
                                  include.space = TRUE,
                                  clustering = TRUE,
                                  seasonPeriod = 1,
                                  equalTimeDist = TRUE,
                                  spatApprox = TRUE,
                                  alphaMethod = "block",
                                  h = 15,
                                  storeSpatPredPara = FALSE,
                                  storeWeights = FALSE,
                                  alphasWeightsToFiles = FALSE)
save(regFixedL.simu.block, file="regFixedL30simuBlockT500M64Iter30000.RData")
Diags.block <- diagnostics(regFixedL.simu.block,
                           diags = c("dic", "dinf", "meanIC", "waic"),
                           keepDeviance = TRUE)
save(Diags.block, file = "regFixedL30simuBlockT500M64Iter30000Diags.RData")
Deviance.block <- as.mcmc(Diags.block$deviance)
save(Deviance.block, file = "regFixedL30simuBlockT500M64Iter30000Deviance.RData")
GibbsStepTimeFixedLblock.fast <- regFixedL.simu.block$GibbsStepTime
save(GibbsStepTimeFixedLblock.fast, file = "GibbsStepTimeFixedLblockFast.RData")
regFixedL.simu.block <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                  starting = NULL, hypers = Hypers, tuning = NULL, m
                                  cmc = MCMC,
                                  L = L,
                                  family = "normal",
                                  temporal.structure = "exponential",
                                  spatial.structure = "continuous",
                                  seed = 29,
                                  gamma.shrinkage = TRUE,
                                  include.time = TRUE,
                                  include.space = TRUE,
                                  clustering = TRUE,
                                  seasonPeriod = 1,
                                  equalTimeDist = FALSE,
                                  spatApprox = TRUE,
                                  alphaMethod = "block",
                                  h = 15,
                                  storeSpatPredPara = FALSE,
                                  storeWeights = FALSE,
                                  alphasWeightsToFiles = FALSE)
save(regFixedL.simu.block,
     file="regFixedL30simuBlockT500M64Iter30000specifyEqualTimeDistF.RData")
Diags.block <- diagnostics(regFixedL.simu.block,
                           diags = c("dic", "dinf", "meanIC", "waic"),
                           keepDeviance = TRUE)
save(Diags.block,
     file = "regFixedL30simuBlockT500M64Iter30000specifyEqualTimeDistFDiags.RData")
Deviance.block <- as.mcmc(Diags.block$deviance)
save(Deviance.block,
     file = "regFixedL30simuBlockT500M64Iter30000specifyEqualTimeDistFDeviance.RData")
GibbsStepTimeFixedLblock <- regFixedL.simu.block$GibbsStepTime
save(GibbsStepTimeFixedLblock, file = "GibbsStepTimeFixedLblock.RData")
regFixedL.simu.sequen <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                   starting = NULL, hypers = Hypers, tuning = NULL,
                                   mcmc = MCMC,
                                   L = L,
                                   family = "normal",
                                   temporal.structure = "exponential",
                                   spatial.structure = "continuous",
                                   seed = 29,
                                   gamma.shrinkage = TRUE,
                                   include.time = TRUE,
                                   include.space = TRUE,
                                   clustering = TRUE,
                                   seasonPeriod = 1,
                                   equalTimeDist = TRUE,
                                   spatApprox = TRUE,
                                   alphaMethod = "sequential",
                                   h = 15,
                                   storeSpatPredPara = FALSE,
                                   storeWeights = FALSE,
                                   alphasWeightsToFiles = FALSE)
save(regFixedL.simu.sequen,
     file = "regFixedL30simuSequenT500M64Iter30000nostorealphaweights.RData")
Diags.sequen <- diagnostics(regFixedL.simu.sequen,
                            diags = c("dic", "dinf", "meanIC", "waic"),
                            keepDeviance = TRUE)
save(Diags.sequen,
     file = "regFixedL30simuSequenT500M64Iter30000DiagsNostorealphaweights.RData")
Deviance.sequen <- as.mcmc(Diags.sequen$deviance)
save(Deviance.sequen,
     file = "regFixedL30simuSequenT500M64Iter30000DevianceNostorealphaweights.RData")
GibbsStepTimeFixedLsequen.fast <- regFixedL.simu.sequen$GibbsStepTime
save(GibbsStepTimeFixedLsequen.fast, file = "GibbsStepTimeFixedLsequenFast.RData")
regFixedL.simu.sequen <- bfaFixedL(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                   starting = NULL, hypers = Hypers, tuning = NULL,
                                   mcmc = MCMC,
                                   L = L,
                                   family = "normal",
                                   temporal.structure = "exponential",
                                   spatial.structure = "continuous",
                                   seed = 29,
                                   gamma.shrinkage = TRUE,
                                   include.time = TRUE,
                                   include.space = TRUE,
                                   clustering = TRUE,
                                   seasonPeriod = 1,
                                   equalTimeDist = FALSE,
                                   spatApprox = TRUE,
                                   alphaMethod = "sequential",
                                   h = 15,
                                   storeSpatPredPara = FALSE,
                                   storeWeights = FALSE,
                                   alphasWeightsToFiles = FALSE)
save(regFixedL.simu.sequen, file =
       "regFixedL30simuSequenT500M64Iter30000specifyEqualTimeDistFnostorealphaweights.RData")
Diags.sequen <- diagnostics(regFixedL.simu.sequen,
                            diags = c("dic", "dinf", "meanIC", "waic"),
                            keepDeviance = TRUE)
save(Diags.sequen, file =
       "regFixedL30simuSequenT500M64Iter30000specifyEqualTimeDistFDiagsNostorealphaweights.RData")
Deviance.sequen <- as.mcmc(Diags.sequen$deviance)
save(Deviance.sequen, file =
       "regFixedL30simuSequenT500M64Iter30000specifyEqualTimeDistFDevianceNostorealphaweights.RData")
GibbsStepTimeFixedLsequen <- regFixedL.simu.sequen$GibbsStepTime
save(GibbsStepTimeFixedLsequen, file = "GibbsStepTimeFixedLsequen.RData")
regVaryLj.simu.sequen <- bfaVaryingLjs(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                       LjVec = LjVec,
                                       starting = NULL, hypers = Hypers, tuning = NULL,
                                       mcmc = MCMC,
                                       family = "normal",
                                       temporal.structure = "exponential",
                                       spatial.structure = "continuous",
                                       seed = 29,
                                       gamma.shrinkage = TRUE,
                                       include.time = TRUE,
                                       include.space = TRUE,
                                       seasonPeriod = 1,
```

```r
                                       equalTimeDist = TRUE,
                                       spatApprox = TRUE,
                                       alphaSequen = TRUE,
                                       h = 15,
                                       storeSpatPredPara = FALSE,
                                       storeWeights = FALSE)
save(regVaryLj.simu.sequen,
     file="regVaryLjsimuSequenT500M64Iter30000nostorealphaweight.RData")
Diags.sequenVaryLj <- diagnostics(regVaryLj.simu.sequen,
                                  diags = c("dic", "dinf", "meanIC", "waic"),
                                  keepDeviance = TRUE)
save(Diags.sequenVaryLj,
     file = "regVaryLjsimuSequenT500M64Iter30000DiagsNostorealphaweight.RData")
Deviance.sequenVaryLj <- as.mcmc(Diags.sequenVaryLj$deviance)
save(Deviance.sequenVaryLj,
     file = "regVaryLjsimuSequenT500M64Iter30000DevianceNostorealphaweight.RData")
GibbsStepTimeVaryLjSequen.fast <- regVaryLj.simu.sequen$GibbsStepTime
save(GibbsStepTimeVaryLjSequen.fast, file = "GibbsStepTimeVaryLjSequenFast.RData")
regVaryLj.simu.sequen <- bfaVaryingLjs(Y ~ 0, data = dat, dist = D, time = Time,  K = K,
                                       LjVec = LjVec,
                                       starting = NULL, hypers = Hypers, tuning = NULL,
                                       mcmc = MCMC,
                                       family = "normal",
                                       temporal.structure = "exponential",
                                       spatial.structure = "continuous",
                                       seed = 29,
                                       gamma.shrinkage = TRUE,
                                       include.time = TRUE,
                                       include.space = TRUE,
                                       seasonPeriod = 1,
                                       equalTimeDist = FALSE,
                                       spatApprox = TRUE,
                                       alphaSequen = TRUE,
                                       h = 15,
                                       storeSpatPredPara = FALSE,
                                       storeWeights = FALSE)
save(regVaryLj.simu.sequen, file =
       "regVaryLjsimuSequenT500M64Iter30000specifyEqualTimeDistFnostorealphaweight.RData")
Diags.sequenVaryLj <- diagnostics(regVaryLj.simu.sequen,
                                  diags = c("dic", "dinf", "meanIC", "waic"),
                                  keepDeviance = TRUE)
save(Diags.sequenVaryLj, file =
       "regVaryLjsimuSequenT500M64Iter30000specifyEqualTimeDistFDiagsNostorealphaweight.RData")
Deviance.sequenVaryLj <- as.mcmc(Diags.sequenVaryLj$deviance)
save(Deviance.sequenVaryLj, file =
       "regVaryLjsimuSequenT500M64Iter30000specifyEqualTimeDistFDevianceNostorealphaweight.RData")
GibbsStepTimeVaryLjSequen <- regVaryLj.simu.sequen$GibbsStepTime
save(GibbsStepTimeVaryLjSequen, file = "GibbsStepTimeVaryLjSequen.RData")
```