```
## INCREASE THE CELL WIDTH FOR BIGGER FIGURES
html"""
<style>
    main {
        margin: 0 auto;
        max-width: 2000px;
        padding-left: max(160px, 10%);
        padding-right: max(160px, 10%);
    }
  svg {
    width: 100%;
  }
</style>
"""
```

Load required packages (dependencies specified in the `Project.toml` file)

```
begin
    using Pkg
    Pkg.activate("../Project.toml")

    using PolyChaos
    using LinearAlgebra
    using DelimitedFiles
    using StatsPlots

    using Plots.PlotMeasures
    using Printf

    using CSV
    using DataFrames

    using NetCDF
    using Dates
    using LaTeXStrings
    using PlutoUI

    using JLD
end
```

Include script with different helper functions. If script is changed, this cell will have to be rerun for changes to reflect.

```
begin
    using Revise
    include("./gsa_utils.jl")
end
```

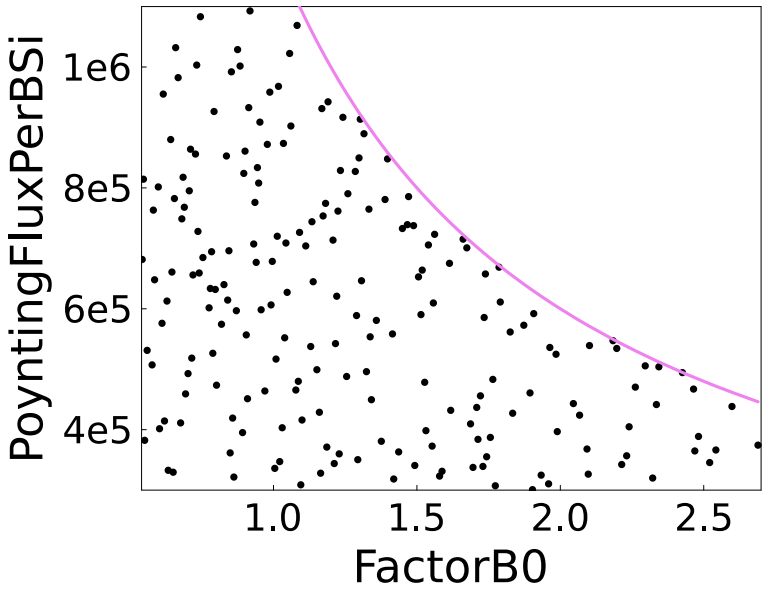
Note: Figures are plotted here in the order that they appear in the manuscript.

Preprocessing

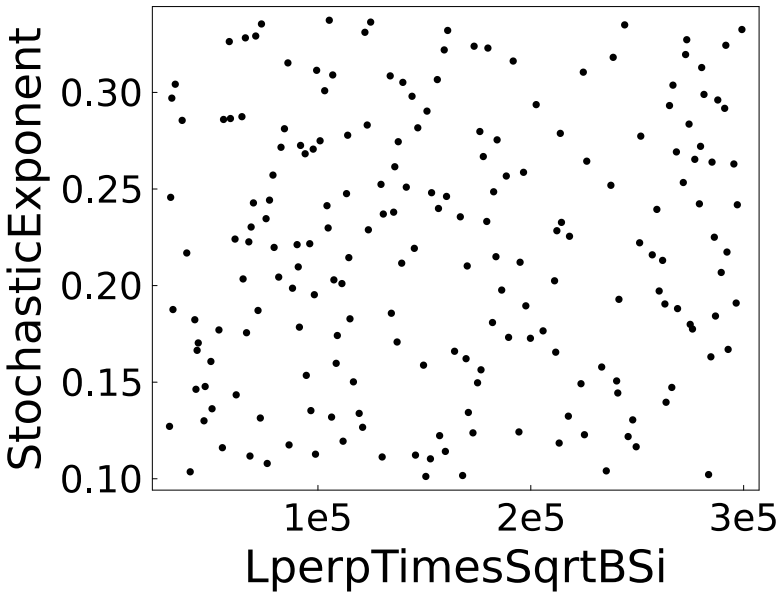
	FactorBo	nChromoSi_AWSoM	PoyntingFluxPerBS
1	1.55304	4.064e17	372800.0
2	0.6156	4.928e18	955200.0
3	0.8964	1.6832e18	824000.0
4	0.66744	2.5472e18	982400.0
5	1.08216	3.9776e18	1.0688e6
6	1.72152	3.5456e18	456000.0
7	0.73656	2.7104e18	728000.0
8	1.29816	2.24e17	849600.0
9	1.0476	3.488e17	627200.0
10	0.5508	3.3728e18	382400.0
⋮ more			
200	1.68696	1.5488e18	409600.0

```
begin
    X_design =
CSV.read("./data/design/X_background_CR2208_update
d.csv", DataFrame)
    pfss = replace(X_design.PFSS, 1=>"HARMONICS",
2=>"FDIPS")
    surfaceWaveRefl =
replace(X_design.UseSurfaceWaveRefl, 1=>"true",
2=>"false")
    select!(X_design, Not([:PFSS,
:UseSurfaceWaveRefl]))
    insertcols!(X_design, :PFSS=>pfss,
:UseSurfaceWaveRefl=>surfaceWaveRefl)
end
```

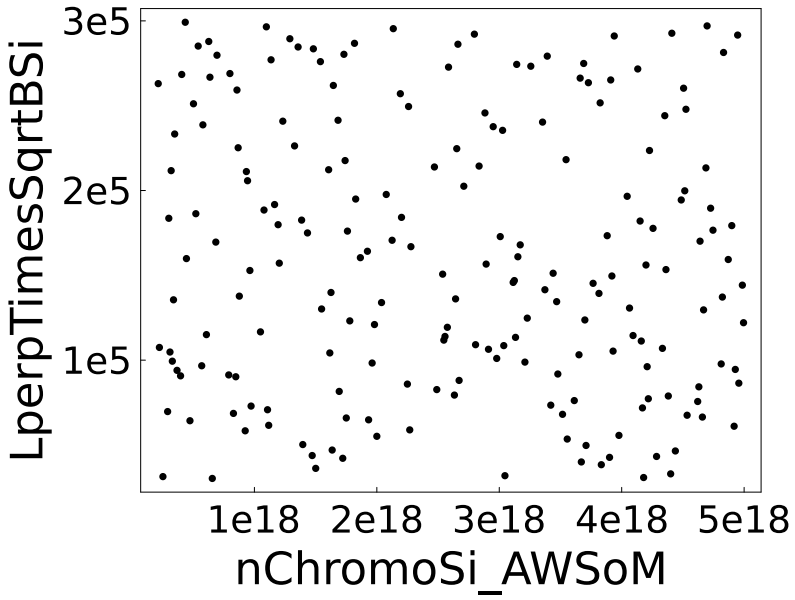
Make selected scatterplots - figure 2 (others can be made in the same fashion by selecting appropriate column names in the plotting arguments)



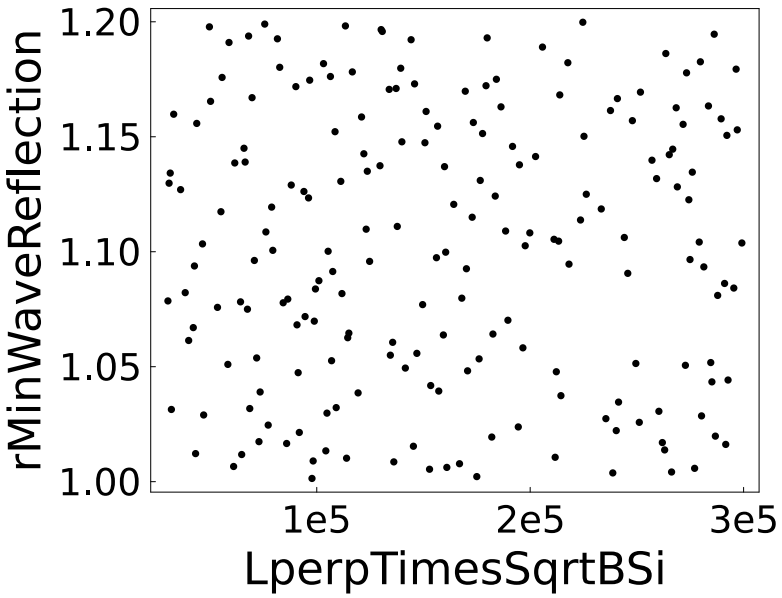

```
begin
    FactorB0PF = scatter(X_design[:, "FactorB0"],
X_design[:, "PoyntingFluxPerBSi"],
                                #
zcolor=shiftWLRMSE.PTRMSE,
                                marker=(:black,
:circle, 4),

xlabel="FactorB0",

ylabel="PoyntingFluxPerBSi",

markerstrokewidth=0,
                                label="",
                                dpi = 300,
                                grid=false
                                )
    plot!(sort(X_design.FactorB0), 1.2e6 ./
(sort(X_design.FactorB0)), line=(:violet, 3.1),
label="")
    plot!(xlims=(0.54, 2.7))
    plot!(ylims=(0.3e6, 1.1e6))
    plot!(guidefontsize=30)
    plot!(tickfontsize=25)
    plot!(framestyle=:box)
    plot!(left_margin=5mm)
    plot!(bottom_margin=5mm)
    plot!(right_margin=8mm)
    plot!(yticks=([4e5, 6e5, 8e5, 1e6], ["4e5",
"6e5", "8e5", "1e6"]))
    plot!(size=(800, 600))
end
```




```
begin
    LperpStoch = scatter(X_design[:,
"LperpTimesSqrtBSi"], X_design[:,
"StochasticExponent"],
                                    #
zcolor=shiftWLRMSE.PTRMSE,
                                    marker=
(:black, :circle, 4),

xlabel="LperpTimesSqrtBSi",

ylabel="StochasticExponent",

markerstrokewidth=0,
                                    label="",
                                    dpi = 300,
                                    grid=false
                                    )
    plot!(guidefontsize=30)
    plot!(tickfontsize=25)
    plot!(framestyle=:box)
    plot!(left_margin=5mm)
    plot!(bottom_margin=5mm)
    plot!(right_margin=8mm)
    plot!(xticks=([1e5, 2e5, 3e5], ["1e5", "2e5",
"3e5"]))
    plot!(size=(800, 600))
end
```




```
begin
    chromoLperp = scatter(X_design[:,
"nChromoSi_AWSoM"], X_design[:,
"LperpTimesSqrtBSi"],
                                    #
zcolor=shiftWLRMSE.PTRMSE,
                                    marker=
(:black, :circle, 4),

xlabel="nChromoSi_AWSoM",

ylabel="LperpTimesSqrtBSi",

markerstrokewidth=0,
                                    label="",
                                    dpi = 300,
                                    grid=false
                                    )
    plot!(guidefontsize=30)
    plot!(tickfontsize=25)
    plot!(framestyle=:box)
    plot!(left_margin=5mm)
    plot!(bottom_margin=5mm)
    plot!(right_margin=8mm)
    plot!(xticks=([1e18, 2e18, 3e18, 4e18, 5e18],
["1e18", "2e18", "3e18", "4e18", "5e18"]))
    plot!(yticks=([1e5, 2e5, 3e5], ["1e5", "2e5",
"3e5"]))
    plot!(size=(800, 600))
end
```




```
begin
    LperpRMin = scatter(X_design[:,
"LperpTimesSqrtBSi"], X_design[:,
"rMinWaveReflection"],
                                    #
zcolor=shiftWLRMSE.PTRMSE,
                                    marker=
(:black, :circle, 4),

xlabel="LperpTimesSqrtBSi",

ylabel="rMinWaveReflection",

markerstrokewidth=0,
                                    label="",
                                    dpi = 300,
                                    grid=false
                                    )
    plot!(guidefontsize=30)
    plot!(tickfontsize=25)
    plot!(framestyle=:box)
    plot!(left_margin=5mm)
    plot!(bottom_margin=5mm)
    plot!(right_margin=8mm)
    plot!(xticks=([1e5, 2e5, 3e5], ["1e5", "2e5",
"3e5"]))
    plot!(size=(800, 600))
end
```

```
Enter cell code...
```

```
Enter cell code...
```

Scale input parameters to [0-1] range

	FactorBo	nChromoSi_AWSoM	PoyntingFluxPerBS
1	0.469	0.043	0.091
2	0.035	0.985	0.819
3	0.165	0.309	0.655
4	0.059	0.489	0.853
5	0.251	0.787	0.961
6	0.547	0.697	0.195
7	0.091	0.523	0.535
8	0.351	0.005	0.687
9	0.235	0.031	0.409
10	0.005	0.661	0.103
⋮	more		
200	0.531	0.281	0.137

```
begin
        lbBg = [0.54, 2e17, 0.3e6, 0.3e5, 0.1, 1]
        ubBg = [2.7, 5e18, 1.1e6, 3e5, 0.34, 1.2]

        paramsBgScaled = (X_design[!, 1:6] .-
lbBg') ./ (ubBg' - lbBg')
end
```

```
inputNames =
▸ ["FactorB0", "nChromoSi_AWSoM", "PoyntingFluxPerBSi",
```

```
inputNames = names(paramsBgScaled)
```

Filter out the single run that failed.

```
excludeRunIdx = ▸ [159]
```

```
excludeRunIdx = [159]
```

List all Parameter Combinations for AWSoM that are not used by us.

	FactorBo	nChromoSi_AWSoM	PoyntingFluxPerBSi
1	1.25928	8.0e17	790400.0

```
X_design[excludeRunIdx, :]
```

```
finalRuns =
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,
```

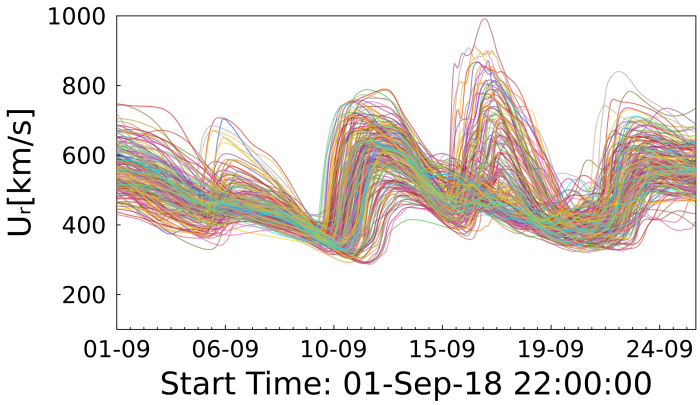
```
finalRuns = setdiff(1:200, excludeRunIdx)
```

Extract QoIs and observations

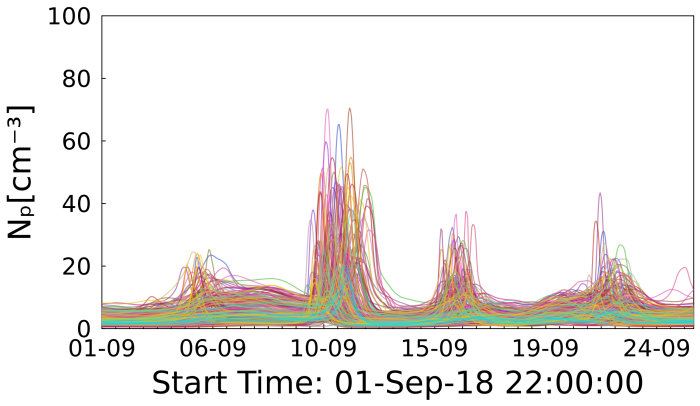
```
[2018-09-01T22:00:00, 2018-09-01T23:00:00, 2018-09-02
```

```
begin
    fn = "./data/background_sims/bg_CR2208.nc"
    UrSim = ncread(fn, "UrSim")
    NpSim = ncread(fn, "NpSim")
    TSim  = ncread(fn, "TSim")
    BSim  = ncread(fn, "BSim")

    UrObs = ncread(fn, "UrObs")
    NpObs = ncread(fn, "NpObs")
    TObs  = ncread(fn, "TObs")
    BObs  = ncread(fn, "BObs")

    startTime = ncgetatt(fn, "time", "startTime")
    timeElapsed = Dates.Hour.(ncread(fn, "time"))
    times = timeElapsed .+
Dates.DateTime(startTime, "yyyy_mm_ddTHH:MM:SS")
end
```

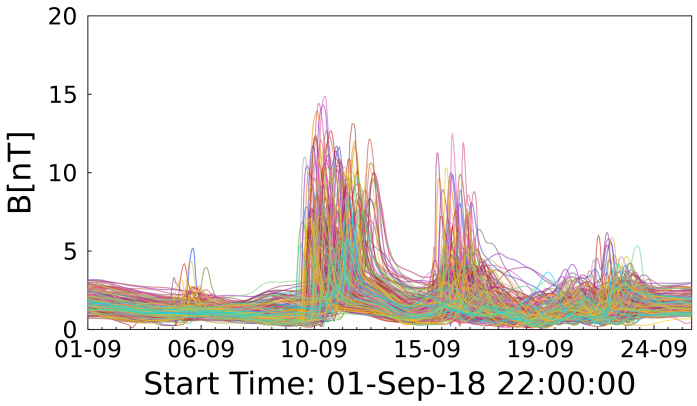
Plot the QoIs (figure 3). If we want to plot observations, supply `plotObs=true` as a keyword.



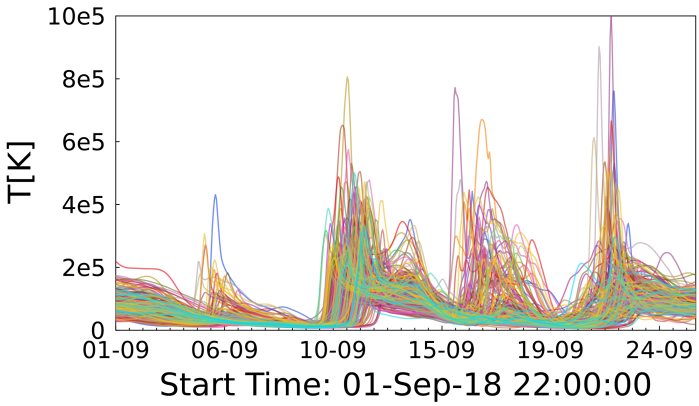

```
begin
    # make plots of the runs that are successful
and satisfy the constraints
    pUrSimObsF = plotSimObs(UrSim, UrObs, times,
collect(1:200); simIdx=finalRuns, plotArgsUr...,
ylims=(100, 1000), dateFormat="dd-mm")
    plot!(size=(920, 470))
    plot!(guidefontsize=24)
    plot!(left_margin=7mm)
    plot!(bottom_margin=7mm)
    plot!(top_margin=5.5mm)
    plot!(ytickfontsize=18)
    plot!(xtickfontsize=18)
    plot!(dpi=300)
end
```




```
begin
    pNpSimObsF = plotSimObs(NpSim, NpObs, times,
collect(1:200); simIdx=finalRuns, plotArgsNp...,
ylims=(0, 100), dateFormat="dd-mm")
    plot!(size=(920, 470))
    plot!(guidefontsize=24)
    plot!(left_margin=7mm)
    plot!(bottom_margin=7mm)
    plot!(top_margin=5.5mm)
    plot!(ytickfontsize=18)
    plot!(xtickfontsize=18)
    plot!(dpi=300)

end
```




```
begin
    pBSimObsF  = plotSimObs(BSim, BObs, times,
collect(1:200); simIdx=finalRuns, plotArgsB...,
dateFormat="dd-mm")
    plot!(size=(920, 470))
    plot!(guidefontsize=24)
    plot!(left_margin=7mm)
    plot!(bottom_margin=7mm)
    plot!(top_margin=5.5mm)
    plot!(ytickfontsize=18)
    plot!(xtickfontsize=18)
    plot!(dpi=300)
end
```




```
begin
    pTSimObsF = plotSimObs(TSim, TObs, times,
collect(1:200); simIdx=finalRuns, plotArgsT...,
ylims=(0, 10e5), dateFormat="dd-mm")
    plot!(yticks=(0:2e5:10e5, [0;
[@sprintf("%de5", i) for i in 2:2:10]][:]))
    plot!(size=(920, 470))
    plot!(guidefontsize=24)
    plot!(left_margin=7mm)
    plot!(bottom_margin=7mm)
    plot!(top_margin=5.5mm)
    plot!(ytickfontsize=18)
    plot!(xtickfontsize=18)
    plot!(dpi=300)
end
```

UQ and GSA

Build surrogate

```
199×6 Matrix{Float64}:
 0.469  0.043  0.091  0.883  0.705  0.813
 0.035  0.985  0.819  0.239  0.223  0.359
 0.165  0.309  0.655  0.783  0.387  0.173
 0.059  0.489  0.853  0.303  0.081  0.409
 0.251  0.787  0.961  0.095  0.775  0.879
 0.547  0.697  0.195  0.697  0.523  0.473
 0.091  0.523  0.535  0.639  0.807  0.707
 ⋮                              ⋮
 0.697  0.829  0.179  0.003  0.607  0.649
 0.481  0.665  0.029  0.923  0.593  0.521
 0.253  0.049  0.225  0.997  0.969  0.519
 0.227  0.293  0.129  0.675  0.535  0.239
 0.387  0.051  0.101  0.481  0.059  0.685
 0.531  0.281  0.137  0.371  0.047  0.983
```

```
XTrainFinal
```

```
199×577 Matrix{Float64}:
 1.56838  1.56981  1.5711   1.57266  1.57443  …  2.037
 3.70051  3.66401  3.62999  3.5992   3.57099     5.754
 2.84923  2.83266  2.81631  2.80147  2.78795     3.064
 3.0836   3.04025  2.99571  2.95295  2.91203     4.583
 5.93575  5.87732  5.82225  5.7701   5.72013     6.883
 4.02272  4.0216   4.02048  4.0207   4.02176  …  4.847
 2.78434  2.7485   2.71371  2.68171  2.6518      2.159
 ⋮                                    ⋱
 6.19138  6.16545  6.14096  6.11934  6.09982     6.399
 2.06734  2.06491  2.06214  2.06022  2.05871     2.637
 1.81644  1.80709  1.79696  1.78674  1.77639  …  2.261
 1.6271   1.60914  1.5904   1.57247  1.5547      2.496
 1.32903  1.32738  1.32612  1.32608  1.32738     1.969
 2.27555  2.28271  2.29066  2.3003   2.31115     2.156
```

```
begin
    XTrainFinal =
Matrix(paramsBgScaled[finalRuns, :])
    YTrainUr = Array{Float64, 2}(UrSim[:,
finalRuns]')
    YTrainNp = Array{Float64, 2}(NpSim[:,
finalRuns]')
end
```

```
ATrainFinal =
199×28 Matrix{Float64}:
 1.0  -0.107387   -1.58309    -1.41682   …  -0.554209
 1.0  -1.61081     1.68009     1.10505       -0.0886064
 1.0  -1.16047    -0.661643    0.536936      -0.94672
 1.0  -1.52767    -0.0381051   1.22283        1.23736
 1.0  -0.862561    0.994197    1.59695       -0.103418
 1.0   0.162813    0.682428   -1.05655   …  -1.11094
 1.0  -1.41682     0.0796743   0.121244       0.146449
 ⋮                                      ⋱    ⋮
 1.0   0.682428    1.13969    -1.11198       -0.96443
 1.0  -0.0658179   0.571577   -1.63159       -1.002
 1.0  -0.855633   -1.56231    -0.952628  …    1.83305
 1.0  -0.9457     -0.717069   -1.28518       -1.1016
 1.0  -0.391443   -1.55538    -1.38218        1.4912
 1.0   0.107387   -0.758638   -1.25747        1.63513
```

```
# build coefficient matrix!
ATrainFinal = buildCoefficientMatrix(XTrainFinal;
pceDegree=2)
```

```
28×577 Matrix{Float64}:
  3.37557      3.36023      3.34484     …   3.80546
  1.54478      1.54815      1.551            1.5647
 -0.0824775   -0.0865056   -0.0902632       -0.0309714
  1.59381      1.57055      1.54717          1.82162
 -0.467954    -0.467728    -0.466839        -0.338891
  0.543254     0.537501     0.531384    …    0.256908
 -0.0274922   -0.0286412   -0.0292732        0.10659
  ⋮                                     ⋱
  0.00704913   0.00724224   0.00764766      -0.0426177
 -0.0611624   -0.0624209   -0.0635754       -0.00122236
  0.0180713    0.0166624    0.015057        -0.0556218
 -0.103302    -0.0998927   -0.0967171   …   -0.132824
 -0.0587152   -0.0606842   -0.0624808       -0.0546192
 -0.0744507   -0.0700515   -0.0656378       -0.232343
```

```
begin
    # Perform ridge regression
    betaUrFinal = solveRegPCE(ATrainFinal,
YTrainUr; λ=lambdaUrFinal)
    betaNpFinal = solveRegPCE(ATrainFinal,
YTrainNp; λ=lambdaNpFinal)
end
```

```
5
```

```
begin
    # set regularization coefficients
    lambdaUrFinal = 0.4
    lambdaNpFinal = 5
end
```

Load matrix of test points

```
XTestFinal =
400×6 Matrix{Float64}:
 0.1335  0.8225  0.4395  0.7605  0.8375  0.1205
 0.5755  0.0015  0.4065  0.0745  0.4725  0.5615
 0.3735  0.4085  0.7355  0.0395  0.5515  0.8215
 0.4875  0.7165  0.5565  0.4925  0.3825  0.0635
 0.6695  0.3005  0.1835  0.1945  0.8175  0.7685
 0.1405  0.3065  0.9965  0.9625  0.8235  0.5935
 0.3895  0.6505  0.0605  0.9155  0.6085  0.4485
 ⋮                                     ⋮
 0.8735  0.6655  0.2345  0.1215  0.6295  0.6125
 0.5365  0.3805  0.0505  0.2995  0.2535  0.6355
 0.2345  0.7785  0.0385  0.7875  0.6615  0.1315
 0.7995  0.7965  0.1015  0.4645  0.0075  0.7175
 0.3695  0.8425  0.6865  0.1525  0.6055  0.4205
 0.1415  0.5565  0.1775  0.4265  0.5505  0.2725
```

```
XTestFinal =
load("./data/design/CR2208TestFinal.jld",
"XTestFinal")[:, 1:6]
```

```
ATest = buildCoefficientMatrix(XTestFinal[:,
1:6]; pceDegree=2);
```

```
577×400 Matrix{Float64}:
 546.309  475.356  502.644  553.63   489.554  …  627.0
 546.096  475.264  502.051  553.86   489.455     626.8
 545.891  475.184  501.495  554.126  489.354     626.7
 545.574  475.084  500.937  554.337  489.264     626.6
 545.199  474.98   500.378  554.551  489.175     626.5
 544.753  474.873  499.818  554.756  489.089  …  626.4
 544.241  474.763  499.254  554.926  489.002     626.2
   ⋮                                    ⋱
 543.73   479.119  502.162  545.257  481.378     585.8
 542.941  478.291  500.507  545.018  481.168     585.6
 542.255  477.497  498.988  544.81   480.997     585.4
 541.588  476.695  497.565  544.606  480.839     585.3
 541.005  475.887  496.258  544.405  480.673  …  585.3
 540.451  475.102  495.015  544.217  480.52      585.2
```

```
begin
    ATestFinalCoeffs = ATest[:, 2:end]
    ATestFinalFiltered = ATestFinalCoeffs .-
mean(ATestFinalCoeffs, dims=1)
    yPredNpFinal = betaNpFinal[1, :] .+
Matrix((ATestFinalFiltered * betaNpFinal[2:end,
:])');
    yPredUrFinal = betaUrFinal[1, :] .+
Matrix((ATestFinalFiltered * betaUrFinal[2:end,
:])');
end
```

Get mean and standard deviations of Ur and Np

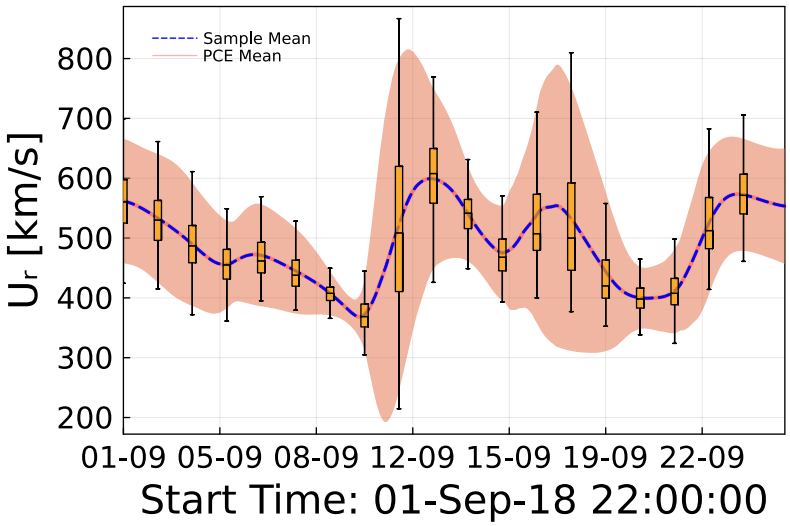
```
▸ [52.0851, 51.9917, 51.9124, 51.8068, 51.7005, 51.5908
```

```
begin
    meanEmpiricalNpFinal = mean(yPredNpFinal;
dims=2)
    stdEmpiricalNpFinal = std(yPredNpFinal;
dims=2)[:]

    meanEmpiricalUrFinal = mean(yPredUrFinal;
dims=2)
    stdEmpiricalUrFinal = std(yPredUrFinal;
dims=2)[:]
end
```

```
577×200 Matrix{Float64}:
 599.652  492.839  596.525  589.894  540.433  …  582.8
 599.058  490.506  596.864  588.817  539.938     583.3
 598.504  488.136  597.066  587.613  539.474     583.8
 597.923  485.728  596.979  586.132  539.006     584.2
 597.345  483.26   596.705  584.435  538.539     584.5
 596.771  480.768  596.242  582.513  538.067  …  584.8
 596.201  478.235  595.619  580.362  537.592     585.0
   ⋮                                        ⋱
 574.008  504.999  596.784  570.987  518.345     556.5
 574.566  505.995  596.813  569.737  516.931     556.8
 575.2    506.753  596.947  568.865  515.637     557.2
 575.858  507.135  597.094  568.284  514.427     557.6
 576.579  507.054  597.359  568.045  513.293  …  558.0
 577.307  506.806  597.631  568.021  512.257     558.3
```

```
UrSim
```

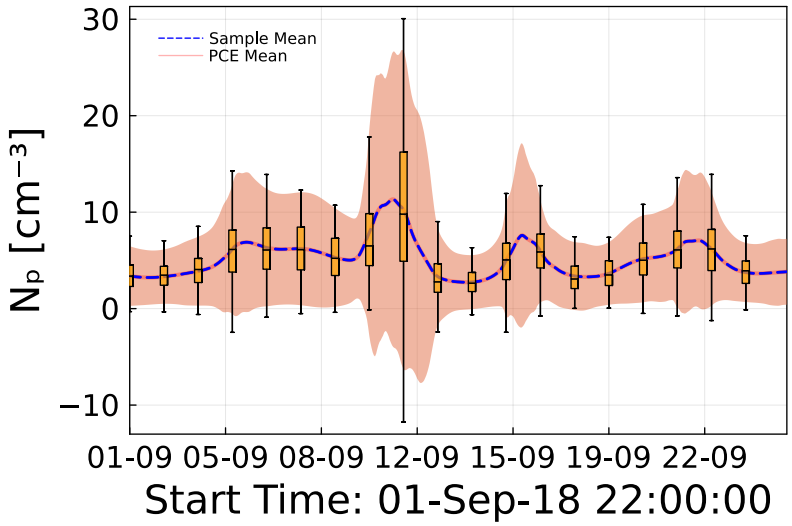
Plot predictive uncertainty (figure 5)




```
begin
    plotUncertainty(yPredUrFinal,
meanEmpiricalUrFinal, stdEmpiricalUrFinal, UrObs,
ylabel="Uᵣ [km/s]", times, nSTD=2, trimIndices=
(1, 577))
    plot!(guidefontsize=20)
    plot!(tickfontsize=15)
    plot!(legend=:topleft)
    plot!(left_margin=3mm)
    plot!(bottom_margin=3mm)
end
```

```
seriestype boxplot has been moved to StatsPlots.  To
ts"); using StatsPlots`
```

```
Keyword argument hover not supported with Plots.GRE
et([:top_margin, :group, :inset_subplots, :backgrou
:yforeground_color_text, :yguidefontcolor, :tickfor
s, :seriesalpha, :seriescolor, :ztick_direction, :x
s, :xtick_direction, :colorbar, :legend_font_family
pha, :overwrite_figure, :arrow, :xguidefonthalign,
xtickfontvalign, :xflip, :zgrid, :fillcolor, :ygrid
cale, :background_color_inside, :zguidefonthalign,
und_color_text, :legend_font_valign, :yscale, :lege
:xgridalpha, :ygridstyle, :clims, :xtickfontcolor,
shape, :background_color_subplot, :ztickfontfamily,
ewidth, :tick_direction, :xguidefontvalign, :xguide
h, :foreground_color_subplot, :xgridlinewidth, :ygu
color, :foreground_color_text, :titlefonthalign, :y
gn, :zgridlinewidth, :ytickfontrotation, :discrete_
:grid, :xguidefontrotation, :ribbon, :xguidefontsiz
oreground_color_axis, :xdiscrete_values, :backgroun
ntcolor, :xgridstyle, :line_z, :size, :orientation,
:markersize, :legend_foreground_color, :camera, :zg
ete_values, :xforeground_color_grid, :seriestype, :
:markerstrokecolor, :ztickfontrotation, :ztickfonth
fontvalign, :xlims, :xforeground_color_border, :mar
:ylink, :levels, :color_palette, :connections, :yfo
s, :zgridstyle, :foreground_color_border, :zguidefo
r_z, :markerstrokealpha, :left_margin, :markeralpha
nnotations, :window_title, :tickfontvalign, :foregr
ontcolor, :ygridlinewidth, :zlink, :zscale, :smooth
ticks, :guidefontsize, :zguidefontsize, :y, :margin
rete_values, :tickfonthalign, :bottom_margin, :yfor
uidefontfamily, :framestyle, :yguidefontvalign, :yg
:zgridalpha, :ztickfontcolor, :scale, :legend_posit
tput_format, :legend_title, :zforeground_color_borc
e, :title, :tickfontcolor, :subplot_index, :flip, :
d_background_color, :tickfontsize, :titlefontvalign
axis, :foreground_color_grid, :xtickfontrotation, :
e, :gridalpha, :xerror, :guidefontfamily, :ylims, :
ntcolor, :primary, :xtickfontfamily, :ytickfontvali
ickfontfamily, :aspect_ratio, :xforeground_color_te
ar_title, :guidefontrotation, :subplot, :label, :yt
uidefontcolor, :yguide, :titlefontsize, :titlefontf
:zforeground_color_axis, :zforeground_color_grid, :
tion, :colorbar_entry, :yguidefontfamily, :polar, :
ries_annotations, :yticks])
```




```
begin
    plotUncertainty(yPredNpFinal,
meanEmpiricalNpFinal, stdEmpiricalNpFinal, NpObs,
ylabel="Nₚ [cm⁻³]", ylims=(-20, 120), times,
nSTD=2, trimIndices=(1, 577))
    plot!(guidefontsize=20)
    plot!(tickfontsize=15)
    plot!(legend=:topleft)
    plot!(left_margin=3mm)
    plot!(bottom_margin=3mm)
end
```

```
seriestype boxplot has been moved to StatsPlots.  To
ts"); using StatsPlots`
```

Sensitivities

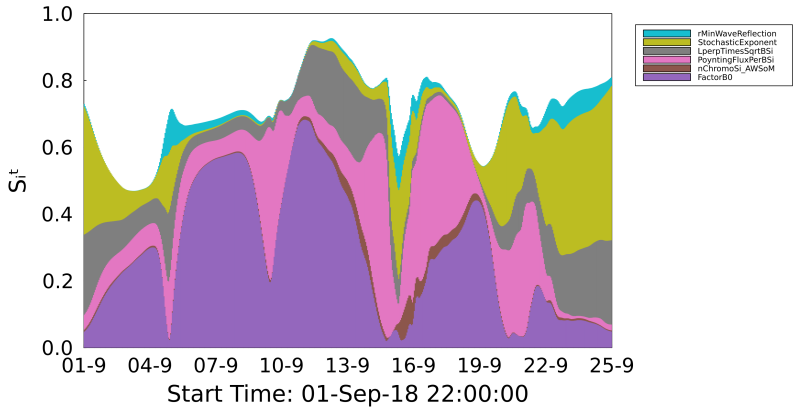
```
[:, :, 3] =
 0.439476     0.00214069   0.00185131   0.00953639
 0.00214069   0.001516     6.18951e-5   0.00101413
 0.00185131   6.18951e-5   0.434686     0.00128597
 0.00953639   0.00101413   0.00128597   0.0395804
 0.00670566   0.000898588  0.00430162   0.000733851
 0.000675754  0.000559978  0.000321792  4.11632e-5

;;; …

[:, :, 575] =
 0.399092    0.00127087   1.59303e-5   0.00856857
 0.00127087  0.00330112   0.00236285   0.000314398
 1.59303e-5  0.00236285   0.537867     2.48759e-5
 0.00856857  0.000314398  2.48759e-5   0.0187911
 0.00132635  5.16197e-5   0.000245465  2.40666e-7
 0.00104362  0.000104693  0.000642755  0.000498317

[:, :, 576] =
 0.398307    0.00124242   8.50009e-7   0.00848664
 0.00124242  0.003492     0.00183688   0.000344232
 8.50009e-7  0.00183688   0.536269     2.36657e-6
 0.00848664  0.000344232  2.36657e-6   0.0201438
 0.00124335  3.02696e-5   8.04075e-5   1.1994e-6
 0.00114179  0.000110669  0.000811963  0.000570869

[:, :, 577] =
 0.396782    0.00120902   2.76108e-5   0.00838526
 0.00120902  0.00366451   0.00139756   0.000374575
 2.76108e-5  0.00139756   0.53537      2.29125e-6
 0.00838526  0.000374575  2.29125e-6   0.0213685
 0.00116705  1.51874e-5   6.06916e-6   2.47475e-6
```

```
begin
    gsaUrFinal = gsa(XTrainFinal, YTrainUr;
regularize=true, pceDegree=2,
lambda=lambdaUrFinal)
    gsaNpFinal = gsa(XTrainFinal, YTrainNp;
regularize=true, pceDegree=2,
lambda=lambdaNpFinal)
end
```

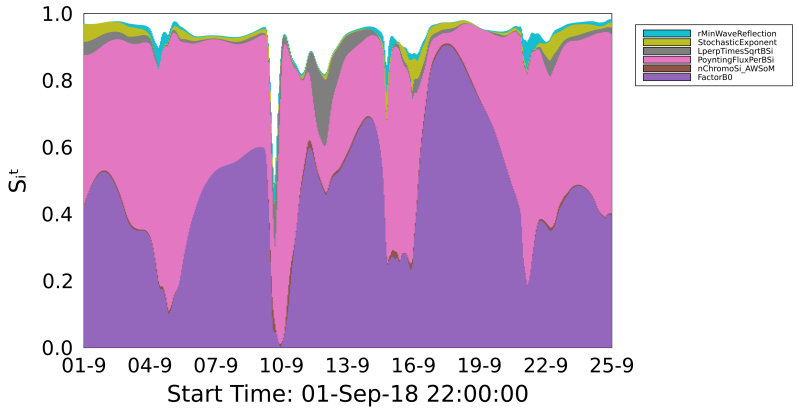
```
6×577 Matrix{Float64}:
 0.425046    0.432231    0.439476     …  0.399092    0
 0.00128505  0.00139725  0.001516        0.00330112  0
 0.449561    0.442053    0.434686        0.537867    0
 0.0387634   0.0392145   0.0395804       0.0187911   0
 0.0541189   0.0535623   0.0529665       0.0134726   0
 0.00111473  0.00102641  0.000937822  …  0.0105251   0
```

```
# Extract main effects
begin
    gsaMainUrFinal =
processMainEffects(gsaUrFinal)
    gsaMainNpFinal =
processMainEffects(gsaNpFinal)
end
```

Plot sensitivities (figures 6 and 7)

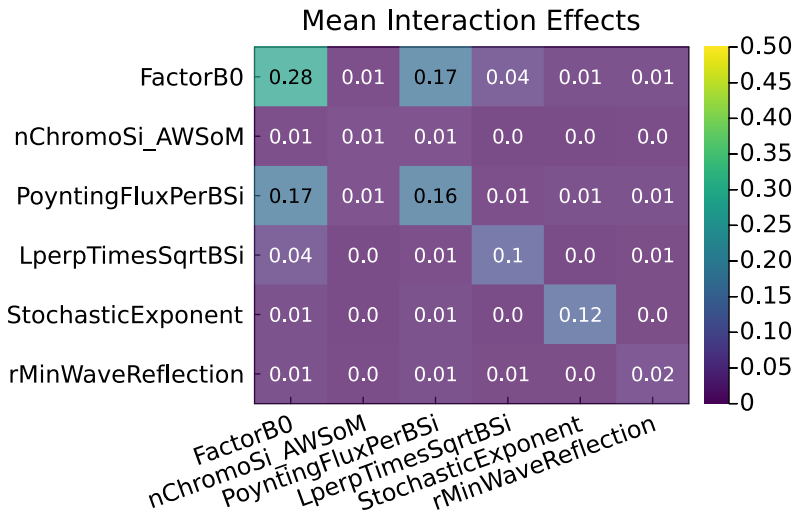



```
begin
    pMainUrFinal =
plotMainEffects(gsaMainUrFinal, times,
inputNames; title = "", dpi=300,
actualStartTime=startTime, tickFormat="dd-m")
    plot!(grid=false)
    plot!(ylabel="S_i ^t")
    plot!(titlefontsize=17)
    plot!(xtickfontsize=20)
    plot!(ytickfontsize=20)
    plot!(guidefontsize=23)
    plot!(leftmargin=7mm)
    plot!(rightmargin=5mm)
    plot!(topmargin=2mm)
    plot!(bottommargin=8mm)
    plot!(size=(1200, 600))
end
```




```
begin
    pMainNpFinal =
plotMainEffects(gsaMainNpFinal, times,
inputNames; title="", dpi=300,
actualStartTime=startTime, tickFormat="dd-m")
    plot!(grid=false)
    plot!(ylabel="Sᵢᵗ")
    plot!(titlefontsize=17)
    plot!(xtickfontsize=20)
    plot!(ytickfontsize=20)
    plot!(guidefontsize=23)
    plot!(leftmargin=7mm)
    plot!(rightmargin=5mm)
    plot!(topmargin=2mm)
    plot!(bottommargin=8mm)
    plot!(size=(1200, 600))
end
```

```
begin
    savefig(pMainUrFinal,
"/Users/ajivani/Downloads/pMainUrFinal2208.png")
    savefig(pMainNpFinal,
"/Users/ajivani/Downloads/pMainNpFinal2208.png")

end
```




```
begin
    pUrIntMeanFinal =
plotInteractionEffects(gsaUrFinal, times,
inputNames, dpi=300) # symmetric matrix
(interactions are read from either the upper
triangle or the lower triangle)
    plot!(tickfontsize=12)
    plot!(bottom_margin=9mm)
    plot!(right_margin=3mm)
end
```


```
begin
    pNpIntMeanFinal =
plotInteractionEffects(gsaNpFinal, times,
inputNames, dpi=300)
    plot!(tickfontsize=12)
    plot!(bottom_margin=9mm)
    plot!(right_margin=3mm)
end
```

```
Enter cell code...
```

Bootstrapping

Load saved bootstrap data. Performing the bootstrap can be quite slow, so the actual command for getting the data is commented out.

```
# UrBootstrap = bootstrapGSA(XTrainFinal, YTrainUr; regularize=false, nStart=20, nEnd=140, nStep=20)
# NpBootstrap = bootstrapGSA(XTrainFinal, YTrainNp; regularize=false, nStart=20, nEnd=140, nStep=20)
```

```
6×7 Matrix{Float64}:
 0.0377869  0.0499936  0.0352907  0.0318524  0.0266902
 0.0303864  0.0242634  0.0145526  0.0109512  0.0094546
 0.0388023  0.0537739  0.0437808  0.0395591  0.0377325
 0.0338576  0.028058   0.0257935  0.0247166  0.023264
 0.0339777  0.0325109  0.0320861  0.0293868  0.0284943
 0.0308869  0.023488   0.0157503  0.0129514  0.0114316
```

```
begin
    UrBootstrap = load("./data/bootstrapping/bootstrapUr2208.jld", "UrBootstrap")
    avgBootstrapUr = mean(UrBootstrap; dims=2)[:, 1, :, :]
    avgBootstrapRepsUr = mean(avgBootstrapUr; dims=2)[:, 1, :]
    stdBootstrapRepsUr = std(avgBootstrapUr; dims=2)[:, 1, :]
end
```

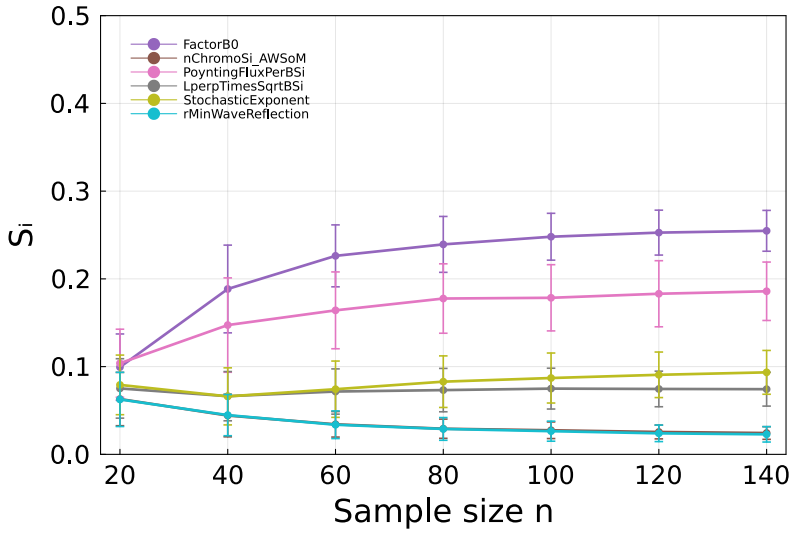
```
6×7 Matrix{Float64}:
 0.0601022  0.044963   0.0217671   0.0159067   0.01405
 0.0298653  0.012393   0.00543184  0.00340099  0.00254
 0.0568441  0.0429031  0.0233962   0.0184295   0.01562
 0.0322113  0.0160089  0.00678868  0.00467677  0.00366
 0.033488   0.0163848  0.00855443  0.00605315  0.00530
 0.031649   0.0132679  0.00565478  0.00428092  0.00298
```

```
begin
    NpBootstrap = load("./data/bootstrapping/bootstrapNp2208.jld", "NpBootstrap")
    avgBootstrapNp = mean(NpBootstrap; dims=2)[:, 1, :, :]
    avgBootstrapRepsNp = mean(avgBootstrapNp; dims=2)[:, 1, :]
    stdBootstrapRepsNp = std(avgBootstrapNp; dims=2)[:, 1, :]
end
```

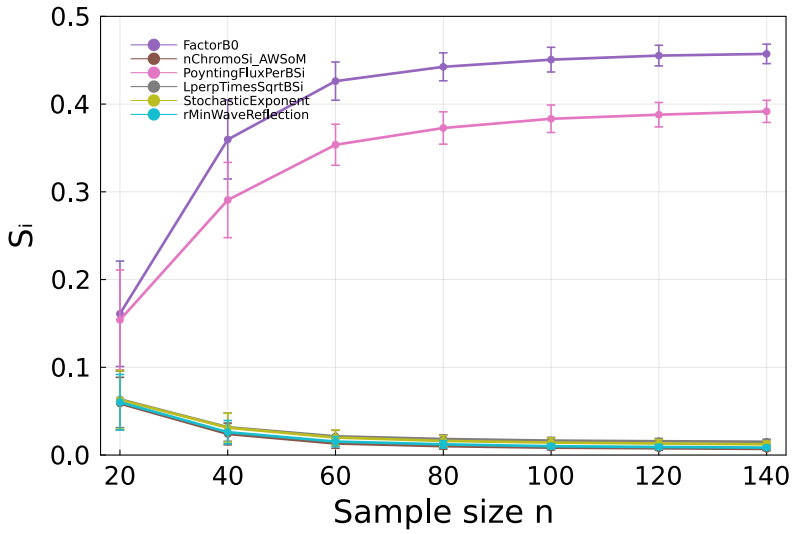
summaryColors =

```
summaryColors = palette(:tab10, rev=true)[6:-1:1]
```

Plot summary of bootstrapping results. (figure 9)




```
begin
    pLineSummaryUr = plot()
    for (idx, eachName) in enumerate(inputNames)
        meanTrend = avgBootstrapRepsUr[idx, :]
        errTrend  = stdBootstrapRepsUr[idx, :]
        plot!(20:20:140, meanTrend,
            yerr=errTrend,
            linewidth=2.5,
            linecolor=summaryColors[idx],
            marker=:circle,
            markercolor=summaryColors[idx],
            markerstrokecolor=summaryColors[idx],
            markerstrokewidth=1.5,
            label=eachName)
    end
    plot!(legend=:topleft)
    plot!(xticks=(20:20:140, string.(20:20:140)))
    plot!(xlabel="Sample size n")
    plot!(ylabel="Sᵢ")
    plot!(guidefontsize=20)
    plot!(tickfontsize=16)
    plot!(leftmargin=5mm)
    plot!(bottommargin=5mm)
    plot!(framestyle=:box)
    plot!(fg_legend=nothing)
    plot!(bg_legend=nothing)
    plot!(size=(750, 500))
    plot!(ylims=(0, 0.5))
end
```




```
begin
    pLineSummaryNp = plot()
    for (idx, eachName) in enumerate(inputNames)
        meanTrend = avgBootstrapRepsNp[idx, :]
        errTrend  = stdBootstrapRepsNp[idx, :]
        plot!(20:20:140, meanTrend,
            yerr=errTrend,
            linewidth=2.5,
            linecolor=summaryColors[idx],
            marker=:circle,
            markercolor=summaryColors[idx],
            markerstrokecolor=summaryColors[idx],
            markerstrokewidth=1.5,
            label=eachName)
    end
    plot!(legend=:topleft)
    plot!(xticks=(20:20:140, string.(20:20:140)))
    plot!(xlabel="Sample size n")
    plot!(ylabel="Sᵢ")
    plot!(guidefontsize=20)
    plot!(tickfontsize=16)
    plot!(leftmargin=5mm)
    plot!(bottommargin=5mm)
    plot!(framestyle=:box)
    plot!(fg_legend=nothing)
    plot!(bg_legend=nothing)
    plot!(size=(750, 500))
    plot!(ylims=(0, 0.5))
end
```

Miscellaneous

```
▶ Dict(:dpi => 200, :ylims => (0, 900000.0), :simWidth
```

```
begin
        plotArgsUr =
Dict(:palette=>:seaborn_bright,
                        :dateFormat=>"dd-mm HH:MM",
                        :tickInterval=>108,
                        # :simIdx => parse.(Int,
EachSimID),
                        :simAlpha=>0.6,
                        :simWidth=>1.2,
                        :ylabel=>"Uᵣ[km/s]",
                        :title=>"",
                        :startTime=>startTime,
                        :dpi=>200,
                        :plotLabels=>:false,

                       )

        plotArgsNp =
Dict(:palette=>:seaborn_bright,
                           :dateFormat=>"dd-mm
HH:MM",
                           :tickInterval=>108,
                           # :simIdx => parse.
(Int, EachSimID),
                           :simAlpha=>0.6,
                           :simWidth=>1.2,
                           :ylabel=>"Nₚ[cm⁻³]",
                           # :ylabel=>"Np [cm" *
L"^{-3}" * "]",
                           :title=>"",
                           :startTime=>startTime,
                           :dpi=>200,
                           :ylims=>(0, 80),
                           :plotLabels=>:false
                          )

        plotArgsB =
Dict(:palette=>:seaborn_bright,
                           :dateFormat=>"dd-mm
HH:MM",
                           :tickInterval=>108,
                           # :simIdx => parse.
(Int, EachSimID),
                           :simAlpha=>0.6,
                           :simWidth=>1.2,
                           :ylabel=>"B[nT]",
                           # :ylabel=>"Np [cm" *
L"^{-3}" * "]",
                           :title=>"",
                           :startTime=>startTime,
                           :dpi=>200,
                           :ylims=>(0, 20),
```

```
                            :plotLabels=>:false,
                            :subtractFactor=>0
                           )

        plotArgsT =
Dict(:palette=>:seaborn_bright,
                           :dateFormat=>"dd-mm HH:MM",
                           :tickInterval=>108,
                           # :simIdx => parse.(Int,
EachSimID),
                           :simAlpha=>0.6,
                           :simWidth=>1.5,
                           :ylabel=>"T[K]",
                           # :ylabel=>"Np [cm" *
L"^{-3}" * "]",
                           :title=>"",
                           :startTime=>startTime,
                           :dpi=>200,
                           :ylims=>(0, 9e5),
                           :plotLabels=>:false,
                           :subtractFactor=>0
                          )
end
```