GlobalSensitivity.jl: Performant and Parallel Global Sensitivity Analysis with Julia

Vaibhav Kumar Dixit and Christopher Rackauckas

1 Julia Computing 2 Massachusetts Institute of Technology 3 Pumas-AI

Summary

Global Sensitivity Analysis (GSA) methods are used to quantify the uncertainty in the output of a model with respect to the parameters. These methods allow practitioners to measure both parameters’ individual contributions and the contribution of their interactions to the output uncertainty. GlobalSensitivity.jl is a Julia (Bezanson et al., 2017) package containing implementations of some of the most popular GSA methods. Currently it supports Delta Moment-Independent (Borgonovo, 2007; Plischke et al., 2013), DGSM (Sobol’ & Kucherenko, 2009), EASI (Plischke, 2010, 2012), eFAST (A. Saltelli et al., 1999; Andrea Saltelli & Bolado, 1998), Morris (Campongle et al., 2007; Morris, 1991), Fractional Factorial (Andrea Saltelli et al., 2008), RBD-FAST (Tarantola et al., 2006), Sobol (Andrea Saltelli, 2002; Andrea Saltelli et al., 2008; Sobol’, 2001) and regression-based sensitivity (Ridolfi & Mooij, 2016) methods.

Examples

The following tutorials in documentation 1 and 2 cover workflows of using GlobalSensitivity.jl on the Lotka-Volterra differential equation, popularly known as the predator-prey model. We present a showcase on how to use multiple GSA methods, analyze their results, and leverage Julia’s parallelism capabilities to perform global sensitivity analysis at scale. The plots have been created using the Makie.jl package (Danisch & Krumbiegel, 2021), while many of the plots in the documentation use the Plots.jl package (Christ et al., 2022).

The ability to introduce parallelism with GlobalSensitivity.jl by using the batch keyword argument is shown in the below code snippet. In the batch interface, each column \( p[:, i] \) is a set of parameters, and we output a column for each set of parameters. Here we present the...
use of Ensemble Interface through EnsembleGPUArray to perform automatic multithreaded
parallelization of the ODE solves.

using GlobalSensitivity, QuasiMonteCarlo, OrdinaryDiffEq

```
function f(du, u, p, t)
end

u0 = [1.0; 1.0]
tspan = (0.0, 10.0)
p = [1.5, 1.0, 3.0, 1.0]
prob = ODEProblem(f, u0, tspan, p)
t = collect(range(0, stop = 10, length = 200))

f1 = function (p)
    prob_func(prob, i, repeat) = remake(prob; p = p[:, i])
    ensemble_prob = EnsembleProblem(prob, prob_func = prob_func)
    sol = solve(ensemble_prob, Tsit5(), EnsembleThreads();
                saveat = t, trajectories = size(p, 2))
    # Now sol[i] is the solution for the i-th set of parameters
    out = zeros(2, size(p, 2))
    for i in 1:size(p, 2)
        out[1, i] = mean(sol[i][1, :])
        out[2, i] = maximum(sol[i][2, :])
    end
    out
end

samples = 10000
lb = [1.0, 1.0, 1.0, 1.0]
ub = [5.0, 5.0, 5.0, 5.0]
sampler = SobolSample()
A,B = QuasiMonteCarlo.generate_design_matrices(samples, lb, ub, sampler)
sobol_result = gsa(f1, Sobol(), A, B, batch=true)
```

**Acknowledgments**

This material is based upon work supported by the National Science Foundation under grant no. OAC-1835443, grant no. SII-2029670, grant no. ECCS-2029670, grant no. OAC-2103804, and grant no. PHY-2021825. We also gratefully acknowledge the U.S. Agency for International Development through Penn State for grant no. S002283-USAID. The information, data, or work presented herein was funded in part by the Advanced Research Projects Agency-Energy (ARPA-E), U.S. Department of Energy, under Award Number DE-AR0001211 and DE-AR0001222. We also gratefully acknowledge the U.S. Agency for International Development through Penn State for grant no. S002283-USAID. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof. This material was supported by The Research Council of Norway and Equinor ASA through Research Council project "308817 - Digital wells for optimal production and drainage". Research was sponsored by the United States Air Force Research Laboratory and the United States Air Force Artificial Intelligence Accelerator and was accomplished under Cooperative Agreement Number

References


