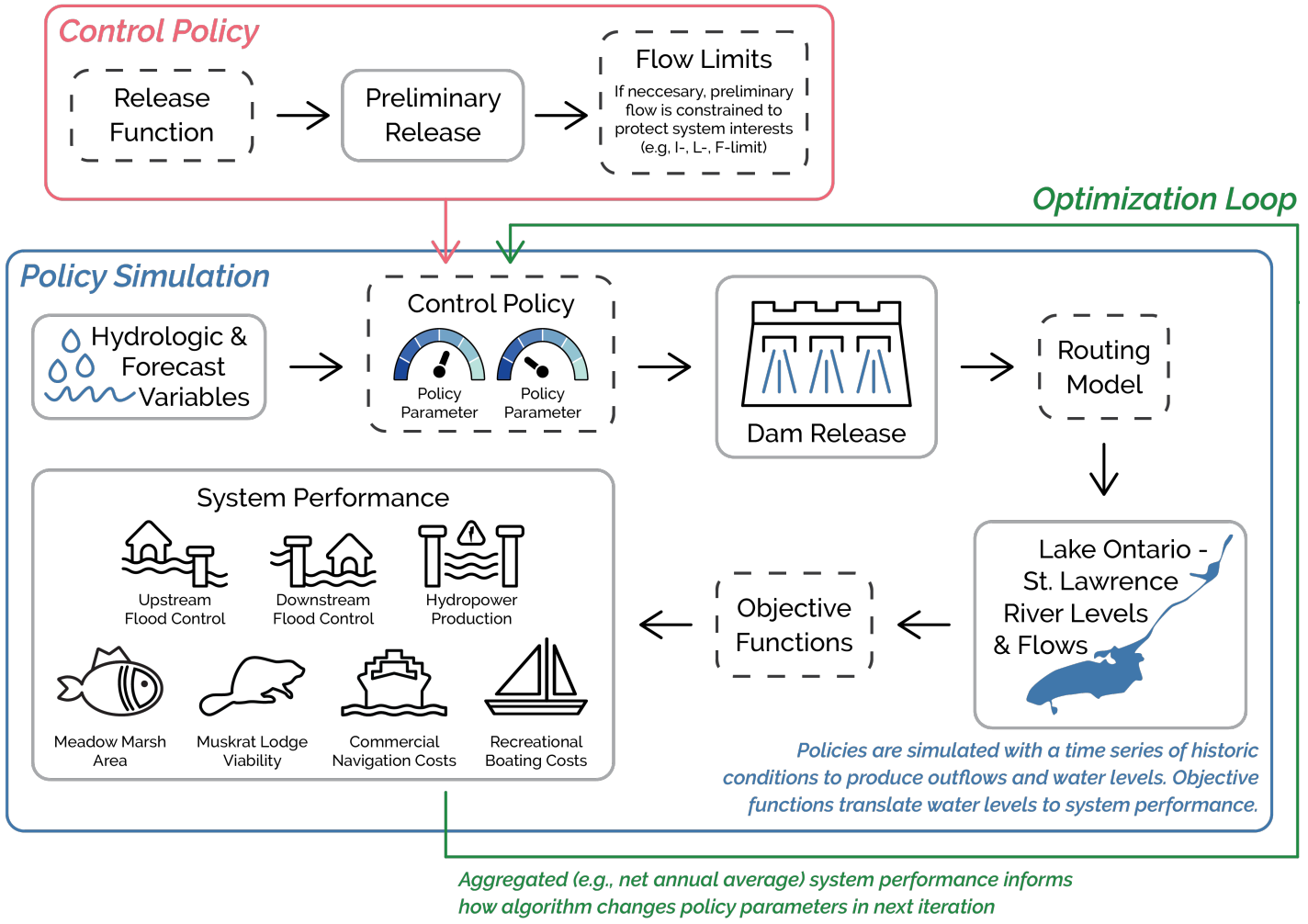
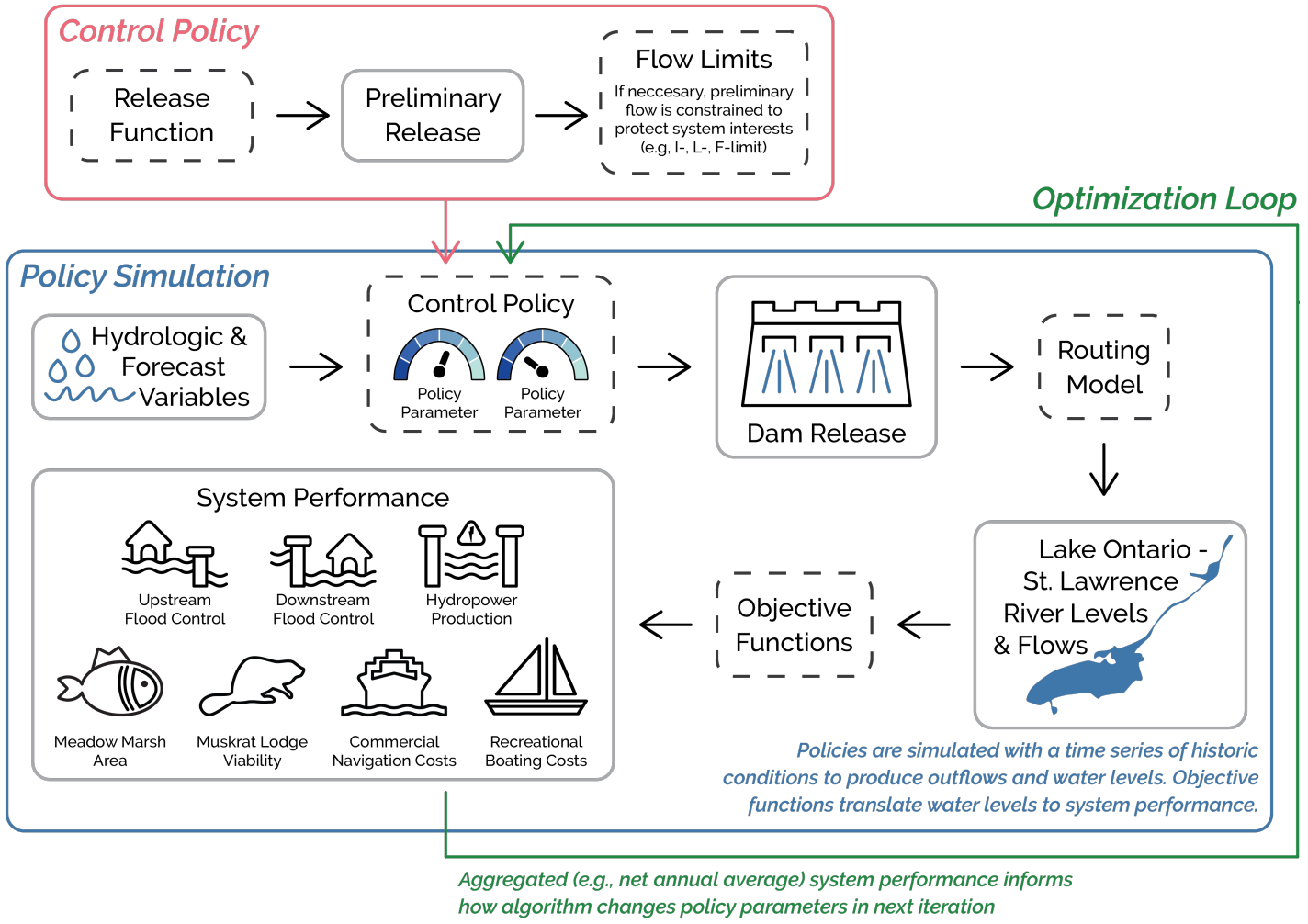
3 major components:

- Simulating a control policy
- Evaluating a control policy's performance
- Optimizing control policy parameters



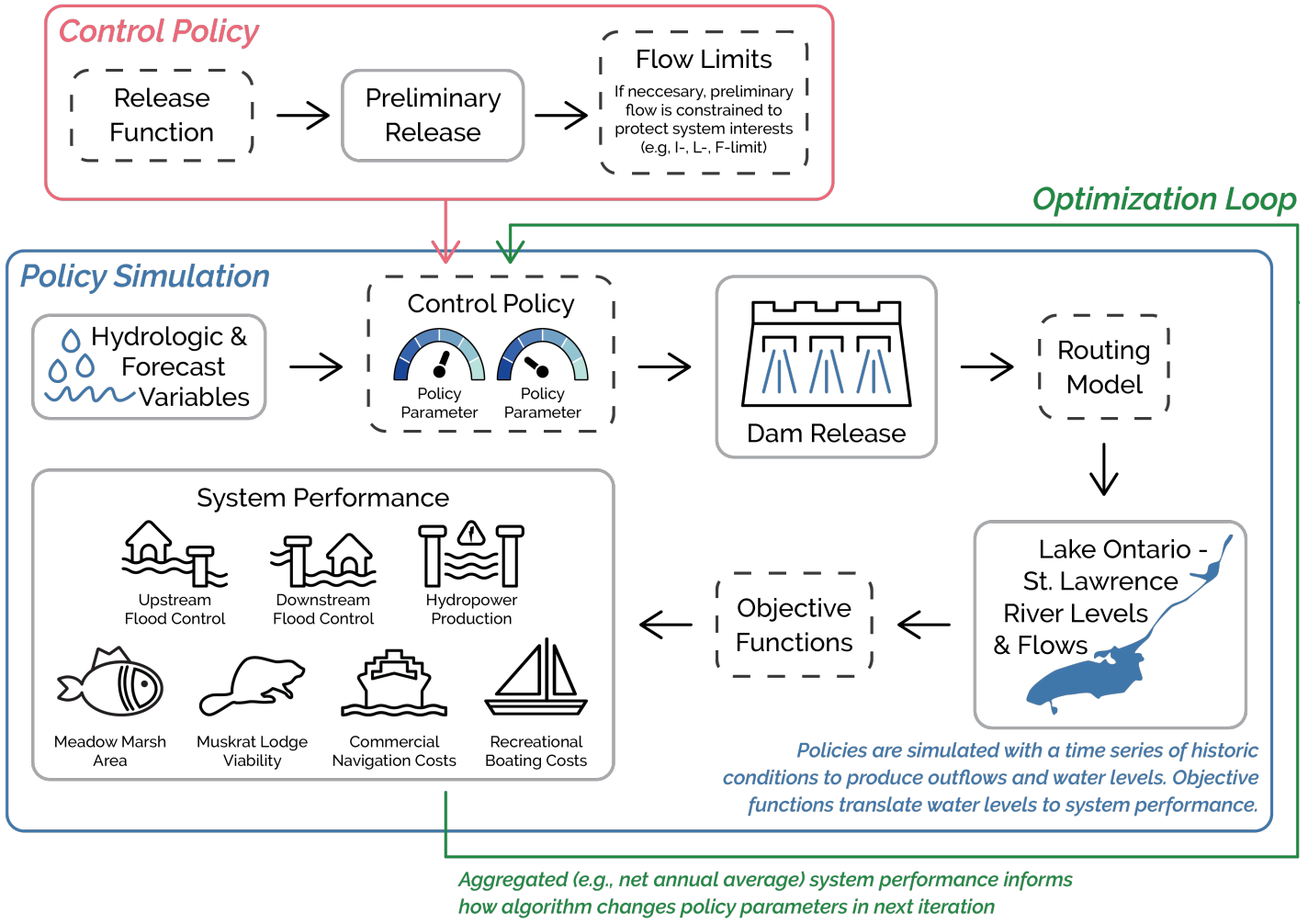
Simulating a control policy

- What is a control policy?
  - A **release function** calculates a preliminary outflow
  - Flow **limit function** adjusts the preliminary outflow (*if necessary*) and returns the final outflow
- Final outflow is **routed** through the system



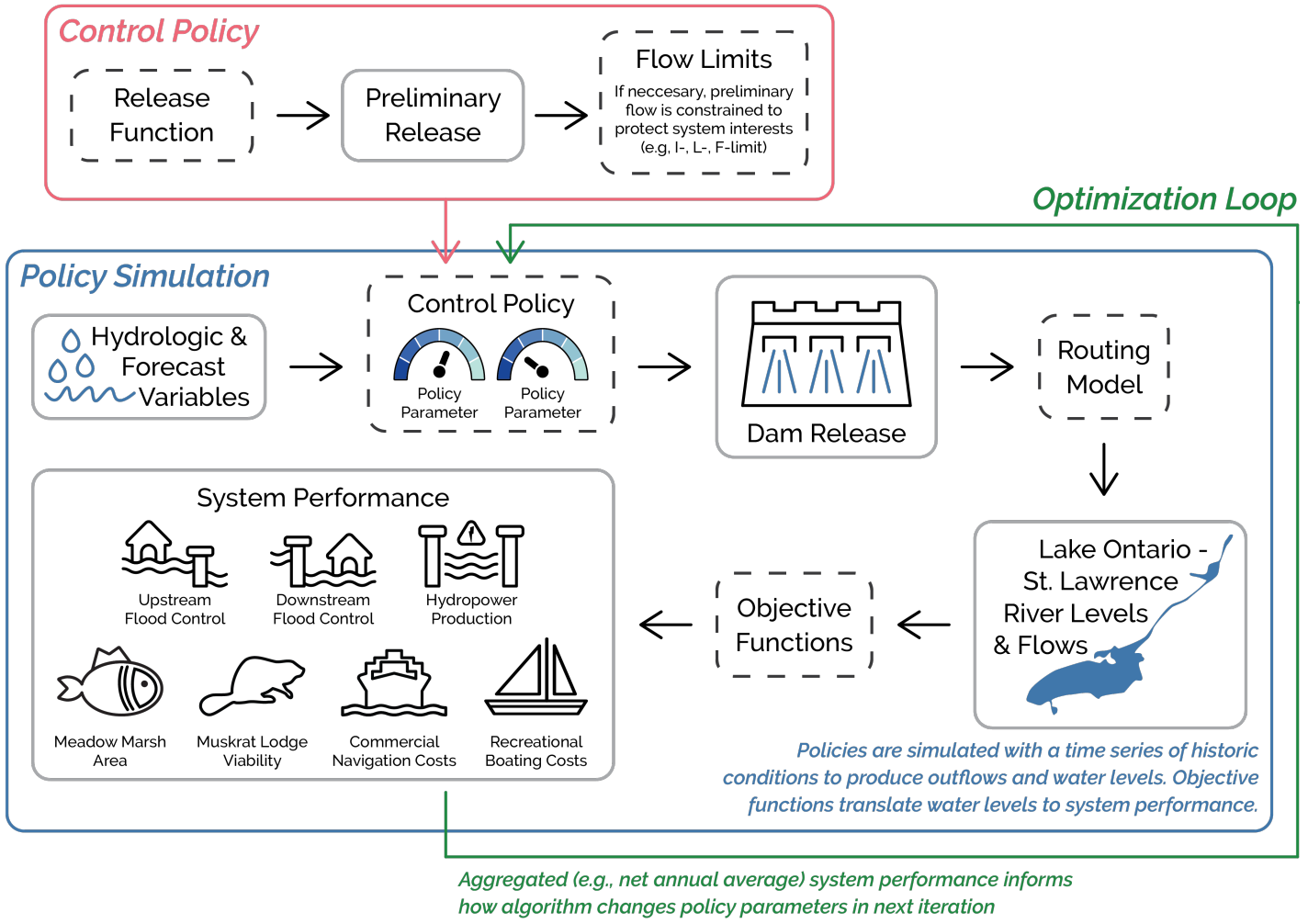
Evaluating a control policy's performance

- Control policy returns a time series of quarter-monthly water levels and flows
- Performance indicator models translate levels and flows into a **time series of system impacts**



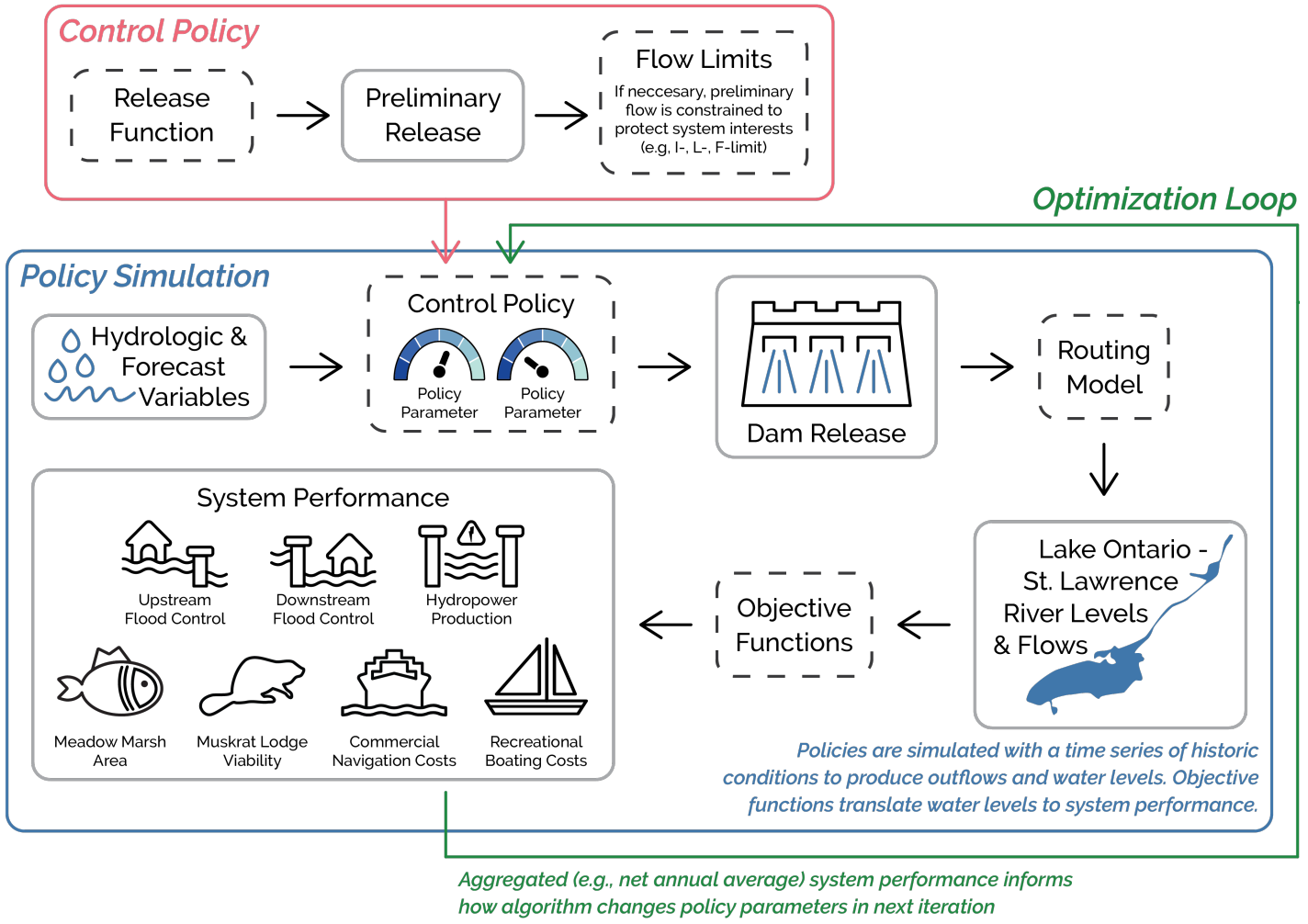
Optimizing control policy parameters

- The time series of system impacts is aggregated to the optimization algorithm
- For upstream flooding, the total number of homes flooded in each year is summed, then the average across all years is returned (i.e. **net annual average**)



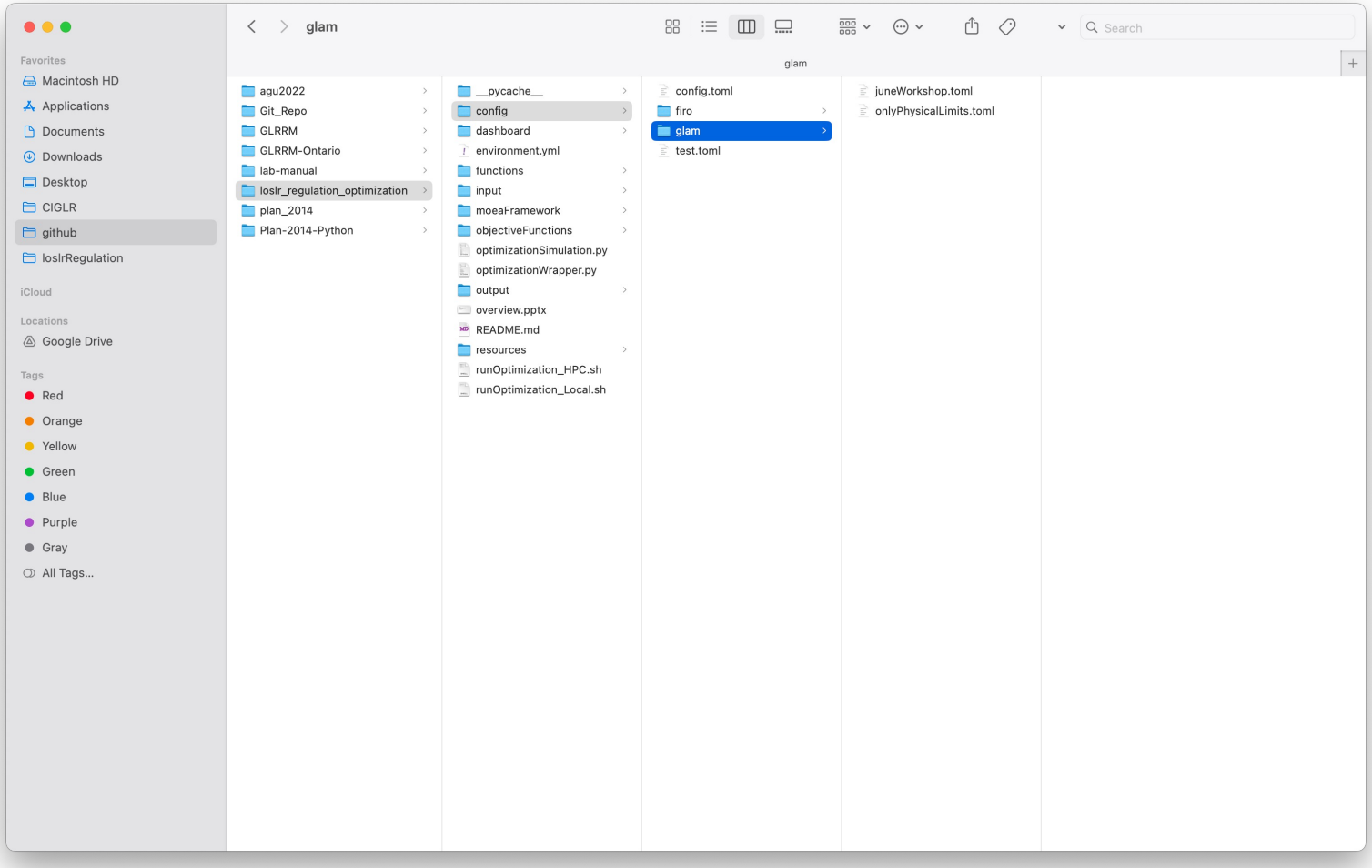
Optimizing control policy parameters

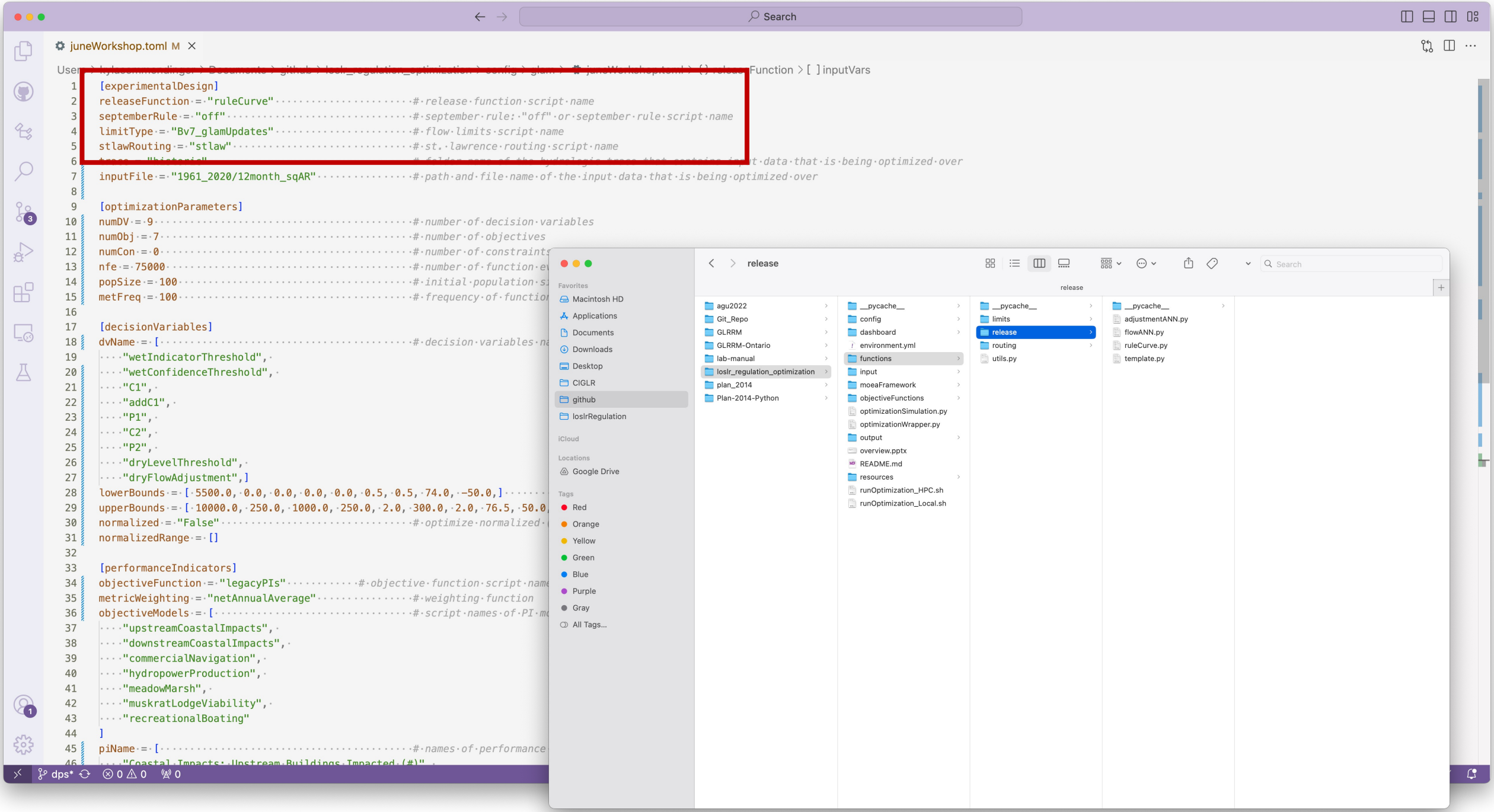
- Optimization algorithm sees 1 value for each objective
- Algorithm uses performance to inform how it changes the dials in the next iteration or function evaluation

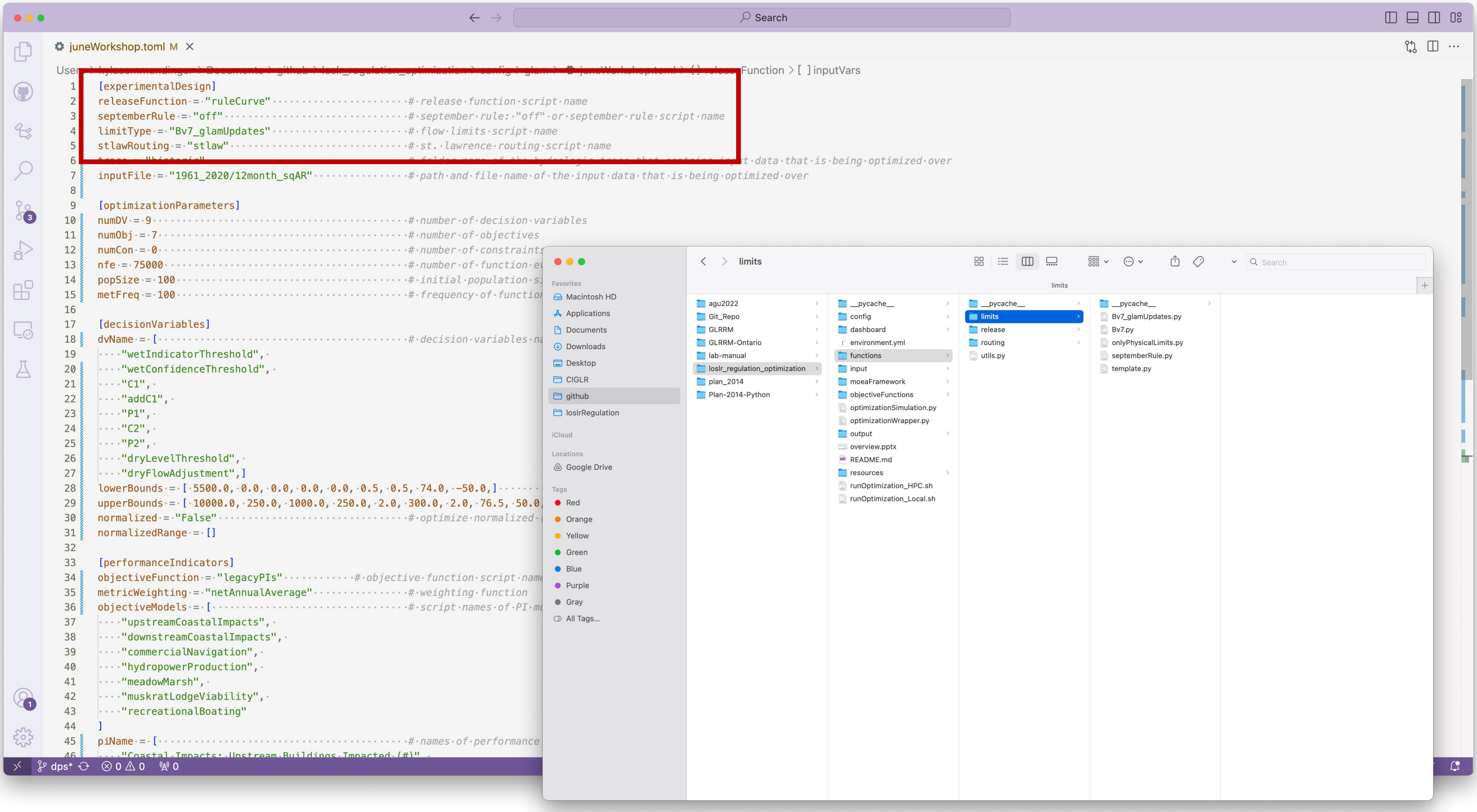


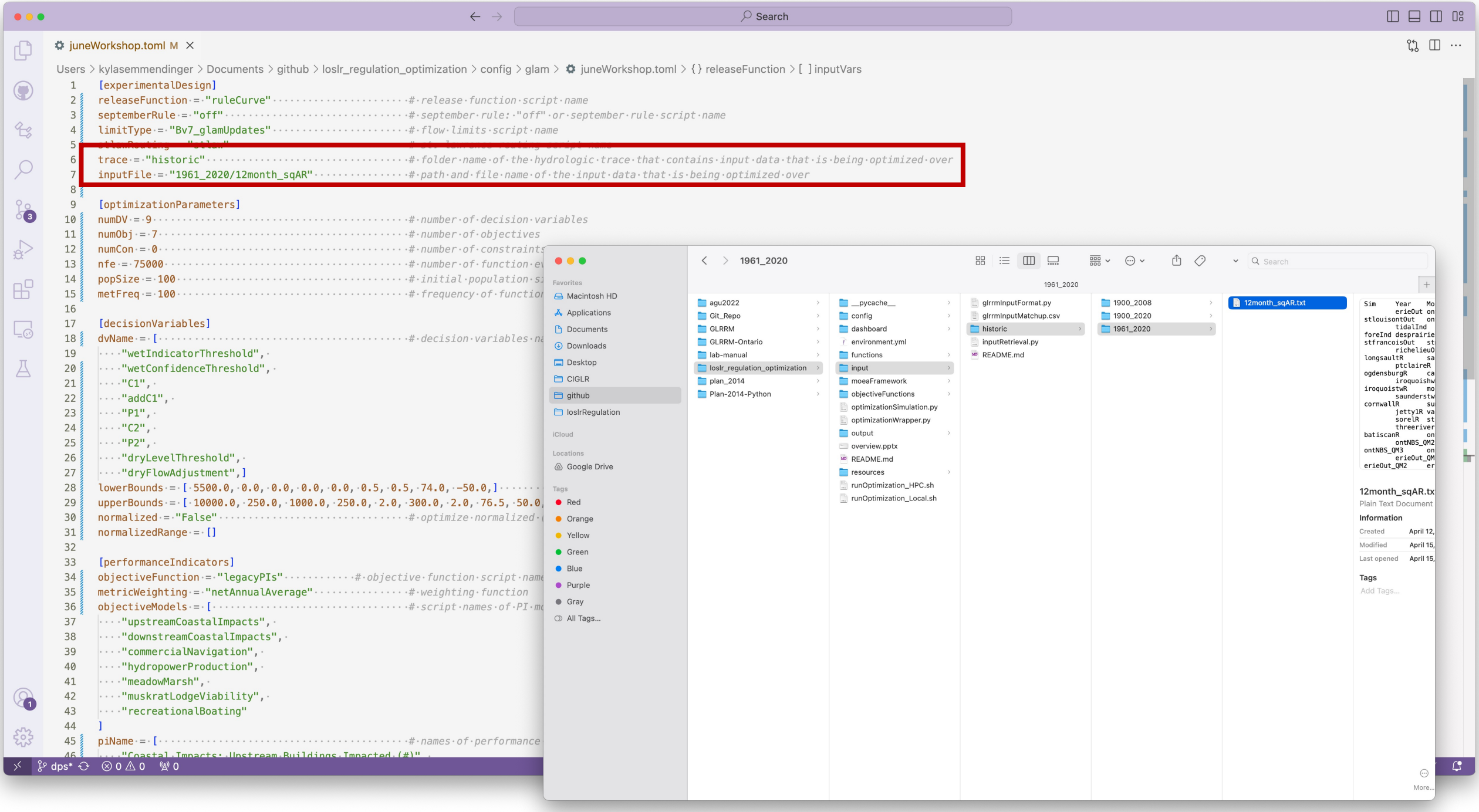
Running an experiment:

- Configuration files setup the experiment and are written in TOML
  - Specify the release function, flow limit function, routing scheme, and objective functions
  - Setup the optimization parameters (e.g. number of function evaluations)



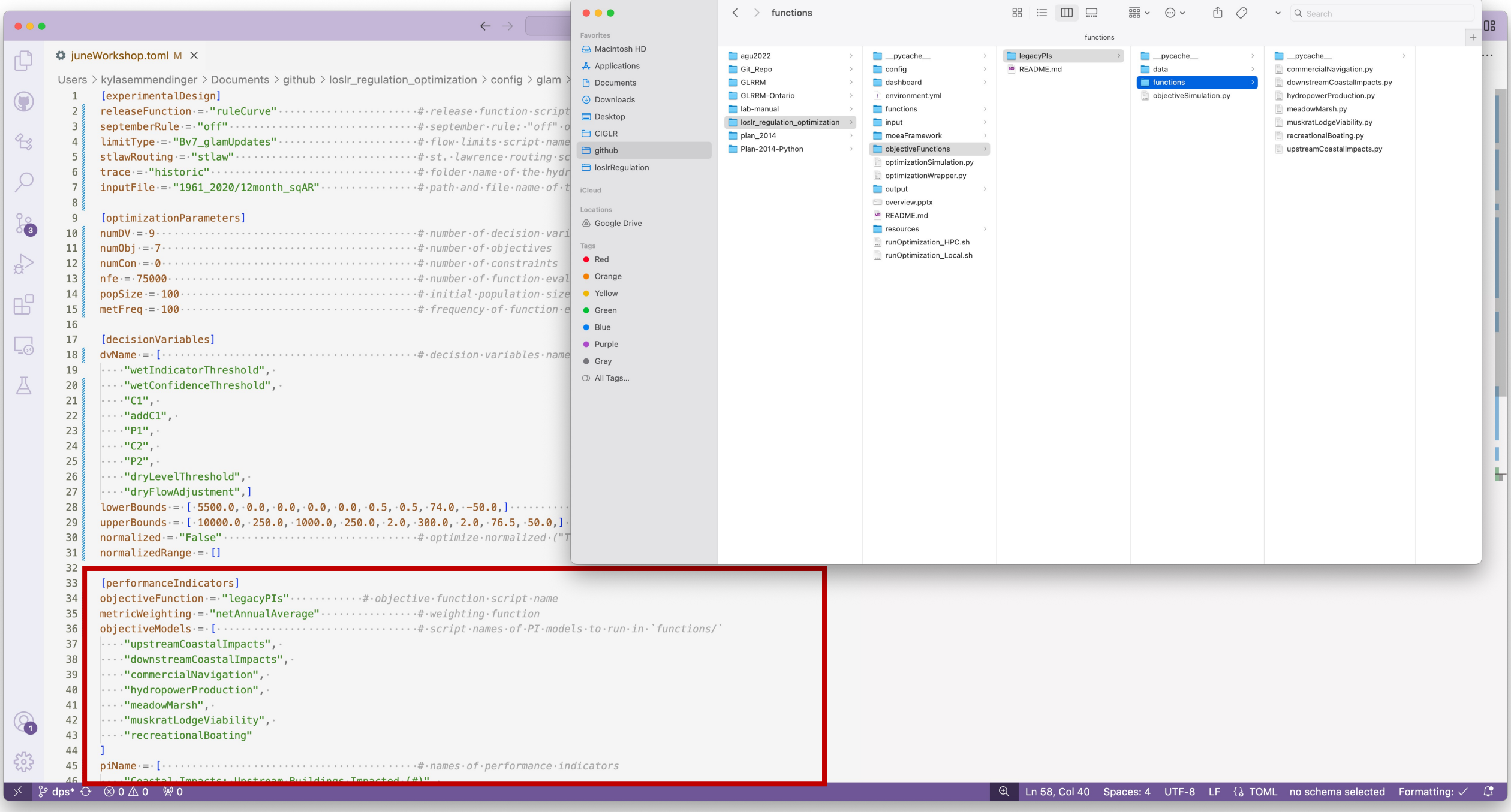

Search
juneWorkshop.toml M
releaseFunction > [ ] inputVars
[experimentalDesign]
releaseFunction = "ruleCurve" #·release·function·script·name
septemberRule = "off" #·september·rule:·"off"·or·september·rule·script·name
limitType = "Bv7_glamUpdates" #·flow·limits·script·name
stlawRouting = "stlaw" #·st.·lawrence·routing·script·name
inputFile = "1961_2020/12month_sqAR" #·path·and·file·name·of·the·input·data·that·is·being·optimized·over
[optimizationParameters]
numDV = 9 #·number·of·decision·variables
numObj = 7 #·number·of·objectives
numCon = 0 #·number·of·constraints
nfe = 75000 #·number·of·function·ev
popSize = 100 #·initial·population·si
metFreq = 100 #·frequency·of·function
[decisionVariables]
dvName = [#·decision·variables·na
"wetIndicatorThreshold",
"wetConfidenceThreshold",
"C1",
"addC1",
"P1",
"C2",
"P2",
"dryLevelThreshold",
"dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,
normalized = "False" #·optimize·normalized·
normalizedRange = []
[performanceIndicators]
objectiveFunction = "legacyPIs" #·objective·function·script·name
metricWeighting = "netAnnualAverage" #·weighting·function
objectiveModels = [#·script·names·of·PI·mo
"upstreamCoastalImpacts",
"downstreamCoastalImpacts",
"commercialNavigation",
"hydropowerProduction",
"meadowMarsh",
"muskratLodgeViability",
"recreationalBoating"
]
piName = [#·names·of·performance
"Coastal Impacts: Upstream Buildings Impacted (#)",
dps*
0 0 0
release
Favorites
Macintosh HD
Applications
Documents
Downloads
Desktop
CIGLR
github
loslrRegulation
iCloud
Locations
Google Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
agu2022
Git_Repo
GLRRM
GLRRM-Ontario
lab-manual
loslr_regulation_optimization
plan_2014
Plan-2014-Python
__pycache__
config
dashboard
environment.yml
functions
input
moeaFramework
objectiveFunctions
optimizationSimulation.py
optimizationWrapper.py
output
overview.pptx
README.md
resources
runOptimization_HPC.sh
runOptimization_Local.sh
__pycache__
limits
release
routing
utils.py
__pycache__
adjustmentANN.py
flowANN.py
ruleCurve.py
template.py


Search
juneWorkshop.toml M
User > lyla > Documents > github > losl_regulation_optimization > config > glam > juneWorkshop.toml > releaseFunction > [ ] inputVars
[experimentalDesign]
releaseFunction = "ruleCurve" #·release·function·script·name
septemberRule = "off" #·september·rule:·"off"·or·september·rule·script·name
limitType = "Bv7_glamUpdates" #·flow·limits·script·name
stlawRouting = "stlaw" #·st.·lawrence·routing·script·name
inputFile = "1961_2020/12month_sqAR" #·path·and·file·name·of·the·input·data·that·is·being·optimized·over
[optimizationParameters]
numDV = 9 #·number·of·decision·variables
numObj = 7 #·number·of·objectives
numCon = 0 #·number·of·constraints
nfe = 75000 #·number·of·function·ev
popSize = 100 #·initial·population·si
metFreq = 100 #·frequency·of·function
[decisionVariables]
dvName = [#·decision·variables·na
"wetIndicatorThreshold",
"wetConfidenceThreshold",
"C1",
"addC1",
"P1",
"C2",
"P2",
"dryLevelThreshold",
"dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,
normalized = "False" #·optimize·normalized·
normalizedRange = []
[performanceIndicators]
objectiveFunction = "legacyPIs" #·objective·function·script·name
metricWeighting = "netAnnualAverage" #·weighting·function
objectiveModels = [#·script·names·of·PI·mo
"upstreamCoastalImpacts",
"downstreamCoastalImpacts",
"commercialNavigation",
"hydropowerProduction",
"meadowMarsh",
"muskratLodgeViability",
"recreationalBoating"
]
piName = [#·names·of·performance
"Coastal Impacts: Upstream Buildings Impacted (#)",
dps*
0 0 0
limits
Favorites
Macintosh HD
Applications
Documents
Downloads
Desktop
CIGLR
github
loslrRegulation
iCloud
Locations
Google Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
agu2022
Git_Repo
GLRRM
GLRRM-Ontario
lab-manual
loslr_regulation_optimization
plan_2014
Plan-2014-Python
__pycache__
config
dashboard
environment.yml
functions
input
moeaFramework
objectiveFunctions
optimizationSimulation.py
optimizationWrapper.py
output
overview.pptx
README.md
resources
runOptimization_HPC.sh
runOptimization_Local.sh
__pycache__
limits
release
routing
utils.py
__pycache__
Bv7_glamUpdates.py
Bv7.py
onlyPhysicalLimits.py
septemberRule.py
template.py

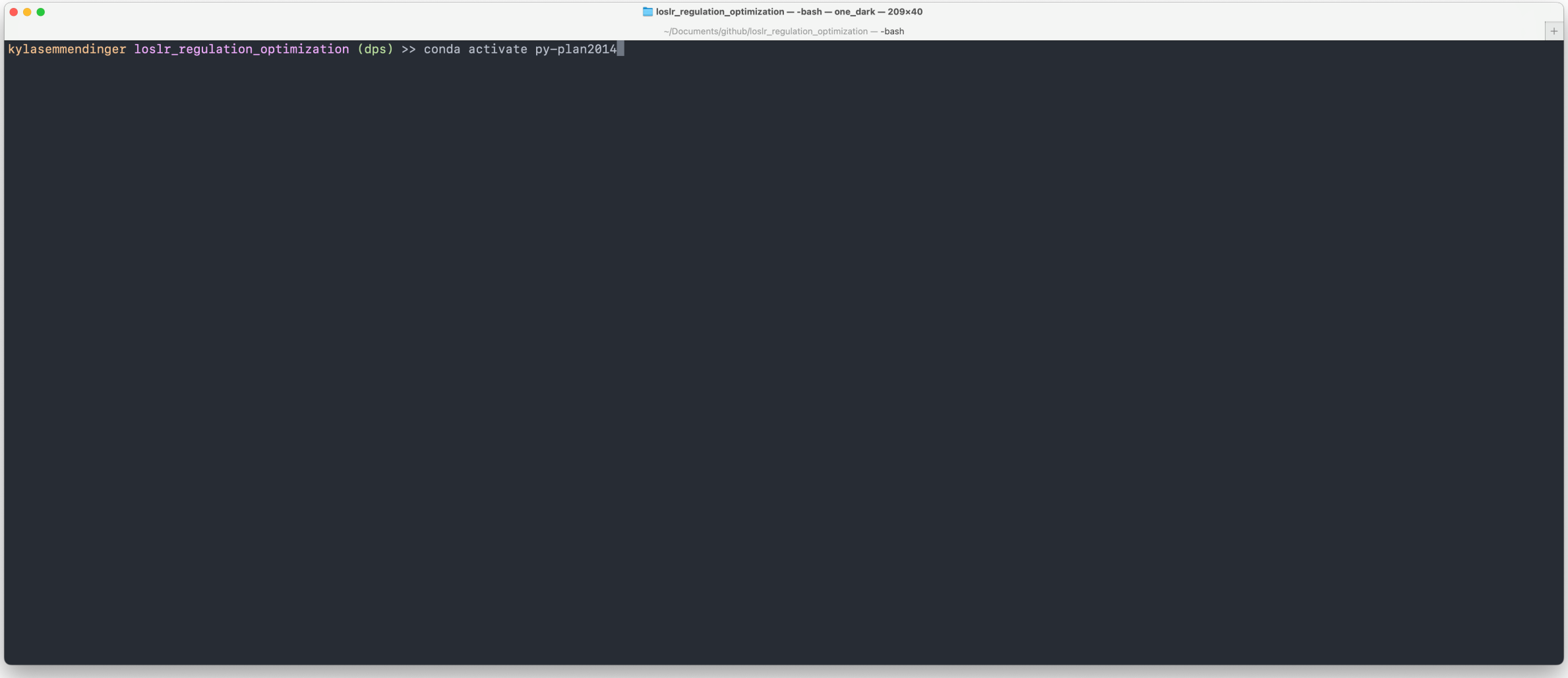

Search
juneWorkshop.toml M
Users > kylasemmendinger > Documents > github > loslr_regulation_optimization > config > glam > juneWorkshop.toml > { } releaseFunction > [ ] inputVars
[experimentalDesign]
releaseFunction = "ruleCurve" #·release·function·script·name
septemberRule = "off" #·september·rule:·"off"·or·september·rule·script·name
limitType = "Bv7_glamUpdates" #·flow·limits·script·name
trace = "historic" #·folder·name·of·the·hydrologic·trace·that·contains·input·data·that·is·being·optimized·over
inputFile = "1961_2020/12month_sqAR" #·path·and·file·name·of·the·input·data·that·is·being·optimized·over
[optimizationParameters]
numDV = 9 #·number·of·decision·variables
numObj = 7 #·number·of·objectives
numCon = 0 #·number·of·constraints
nfe = 75000 #·number·of·function·ev
popSize = 100 #·initial·population·si
metFreq = 100 #·frequency·of·function
[decisionVariables]
dvName = [#·decision·variables·na
"wetIndicatorThreshold",
"wetConfidenceThreshold",
"C1",
"addC1",
"P1",
"C2",
"P2",
"dryLevelThreshold",
"dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,
normalized = "False" #·optimize·normalized·
normalizedRange = []
[performanceIndicators]
objectiveFunction = "legacyPIs" #·objective·function·script·name
metricWeighting = "netAnnualAverage" #·weighting·function
objectiveModels = [#·script·names·of·PI·mo
"upstreamCoastalImpacts",
"downstreamCoastalImpacts",
"commercialNavigation",
"hydropowerProduction",
"meadowMarsh",
"muskratLodgeViability",
"recreationalBoating"
]
piName = [#·names·of·performance
"Coastal Impacts: Upstream Buildings Impacted (#)",
dps*
0 0 0
1961_2020
Favorites
Macintosh HD
Applications
Documents
Downloads
Desktop
CIGLR
github
loslrRegulation
iCloud
Locations
Google Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
agu2022
Git_Repo
GLRRM
GLRRM-Ontario
lab-manual
loslr_regulation_optimization
plan_2014
Plan-2014-Python
__pycache__
config
dashboard
environment.yml
functions
input
moeaFramework
objectiveFunctions
optimizationSimulation.py
optimizationWrapper.py
output
overview.pptx
README.md
resources
runOptimization_HPC.sh
runOptimization_Local.sh
glrrmInputFormat.py
glrrmInputMatchup.csv
historic
inputRetrieval.py
README.md
1900_2008
1900_2020
1961_2020
12month_sqAR.txt
12month_sqAR.tx
Plain Text Document
Information
Created April 12,
Modified April 15,
Last opened April 15,
Tags
Add Tags...
More...

```toml
[experimentalDesign]
releaseFunction = "ruleCurve"                    # release function script name
septemberRule = "off"                            # september rule: "off" or september rule script name
limitType = "Bv7_glamUpdates"                    # flow limits script name
stlawRouting = "stlaw"                           # st. lawrence routing script name
trace = "historic"                               # folder name of the hydrologic trace that contains input data that is being optimized over
inputFile = "1961_2020/12month_sqAR"             # path and file name of the input data that is being optimized over

[optimizationParameters]
numDV = 9                                        # number of decision variables
numObj = 7                                       # number of objectives
numCon = 0                                       # number of constraints
nfe = 75000                                      # number of function evaluations
popSize = 100                                    # initial population size
metFreq = 100                                    # frequency of function evaluations to report metrics

[decisionVariables]
dvName = [                                       # decision variables names
    "wetIndicatorThreshold",
    "wetConfidenceThreshold",
    "C1",
    "addC1",
    "P1",
    "C2",
    "P2",
    "dryLevelThreshold",
    "dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]            # lower bounds of decision variables
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,]    # upper bounds of decision variables
normalized = "False"                             # optimize normalized ("True") or actual value ranges ("False")
normalizedRange = []

[performanceIndicators]
objectiveFunction = "legacyPIs"            # objective function script name
metricWeighting = "netAnnualAverage"               # weighting function
objectiveModels = [                              # script names of PI models to run in `functions/`
    "upstreamCoastalImpacts",
    "downstreamCoastalImpacts",
    "commercialNavigation",
    "hydropowerProduction",
    "meadowMarsh",
    "muskratLodgeViability",
    "recreationalBoating"
]
piName = [                                       # names of performance indicators
    "Coastal Impacts: Upstream Buildings Impacted (#)",
```

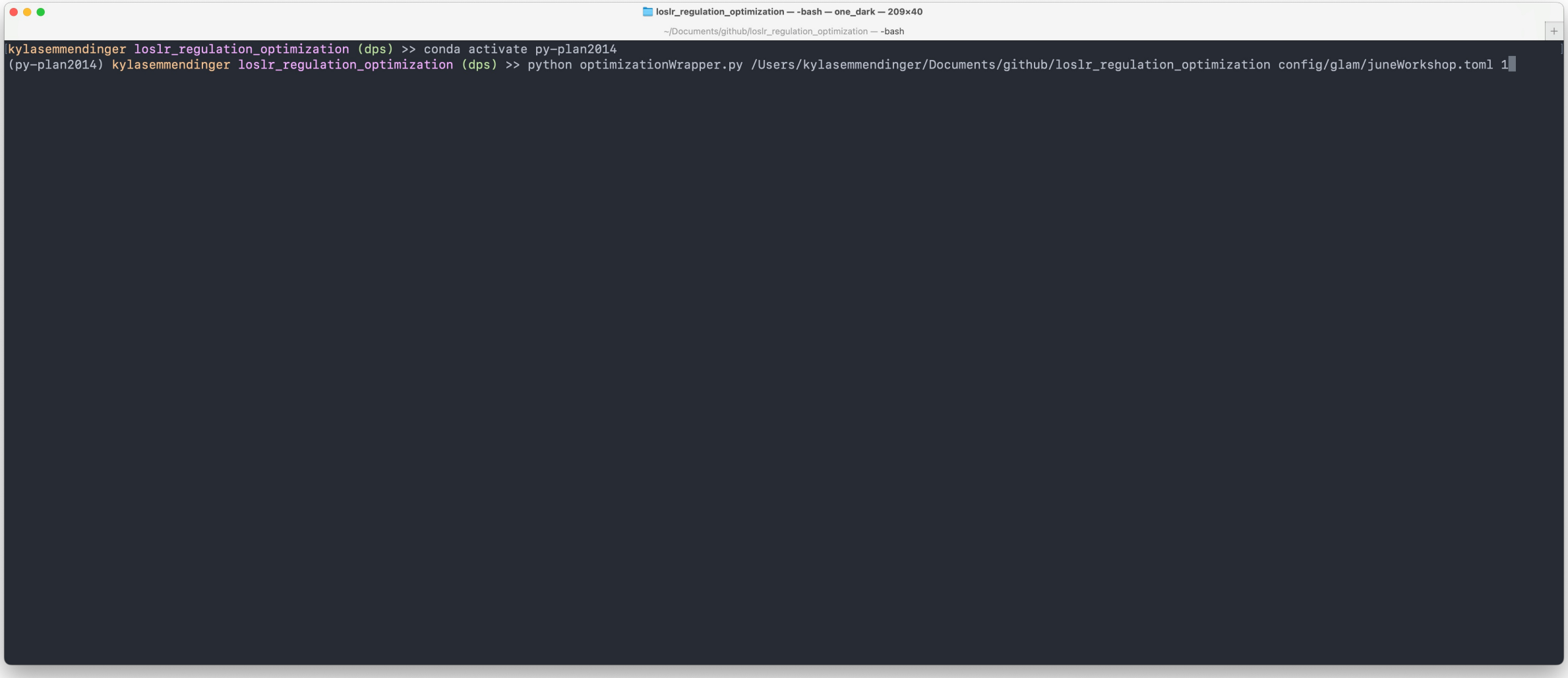
```toml
[experimentalDesign]
releaseFunction = "ruleCurve"                    # release function script name
septemberRule = "off"                            # september rule: "off" or september rule script name
limitType = "Bv7_glamUpdates"                    # flow limits script name
stlawRouting = "stlaw"                           # st. lawrence routing script name
trace = "historic"                               # folder name of the hydrologic trace that contains input data that is being optimized over
inputFile = "1961_2020/12month_sqAR"             # path and file name of the input data that is being optimized over

[optimizationParameters]
numDV = 9                                        # number of decision variables
numObj = 7                                       # number of objectives
numCon = 0                                       # number of constraints
nfe = 75000                                      # number of function evaluations
popSize = 100                                    # initial population size
metFreq = 100                                    # frequency of function evaluations to report metrics

[decisionVariables]
dvName = [                                       # decision variables names
    "wetIndicatorThreshold",
    "wetConfidenceThreshold",
    "C1",
    "addC1",
    "P1",
    "C2",
    "P2",
    "dryLevelThreshold",
    "dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]            # lower bounds of decision variables
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,]    # upper bounds of decision variables
normalized = "False"                             # optimize normalized ("True") or actual value ranges ("False")
normalizedRange = []

[performanceIndicators]
objectiveFunction = "legacyPIs"            # objective function script name
metricWeighting = "netAnnualAverage"       # weighting function
objectiveModels = [                        # script names of PI models to run in `functions/`
    "upstreamCoastalImpacts",
    "downstreamCoastalImpacts",
    "commercialNavigation",
    "hydropowerProduction",
    "meadowMarsh",
    "muskratLodgeViability",
    "recreationalBoating"
]
piName = [                                       # names of performance indicators
    "Coastal Impacts: Upstream Buildings Impacted (#)",
```

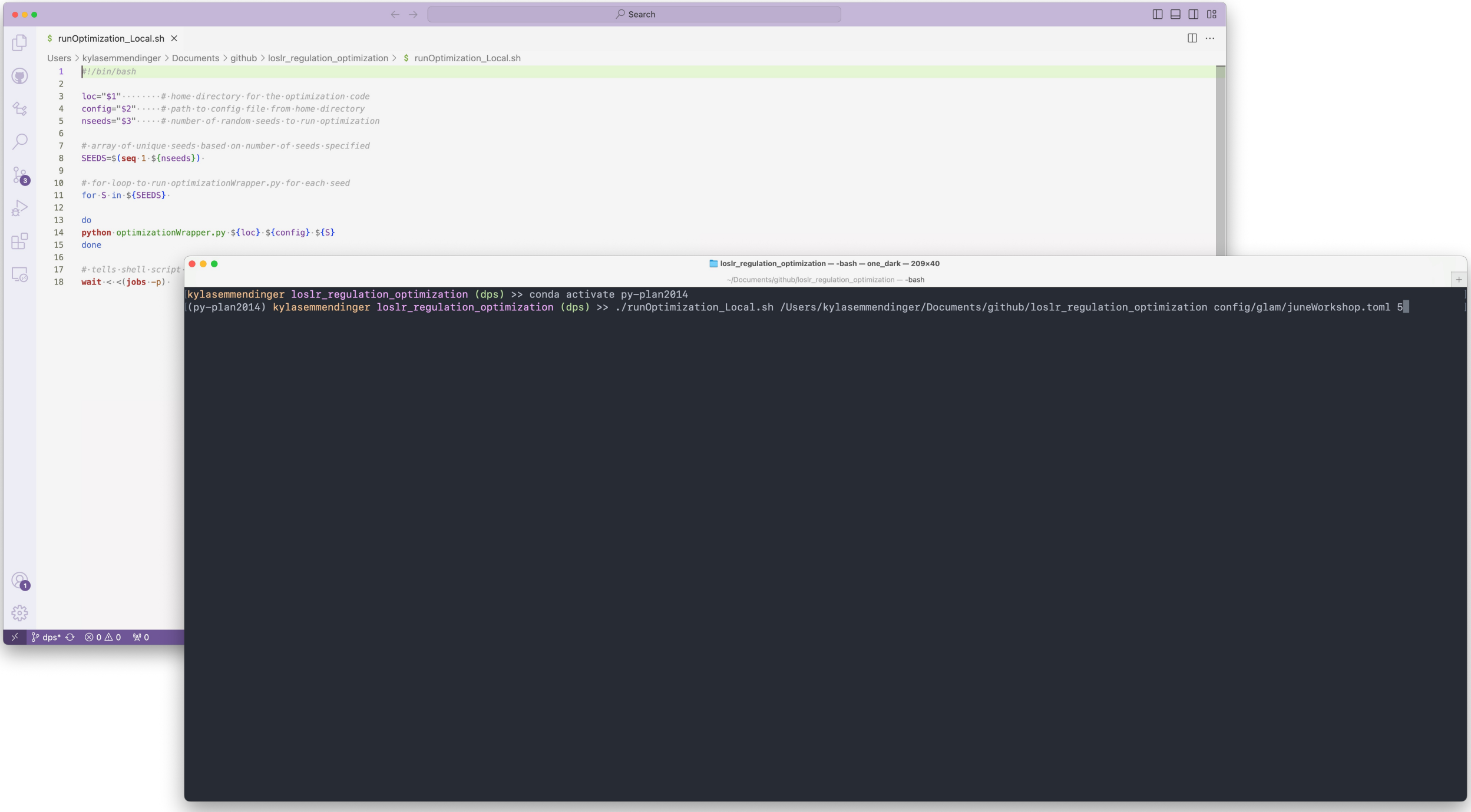

juneWorkshop.toml M
Users > kylasemmendinger > Documents > github > loslr_regulation_optimization > config > glam >
[experimentalDesign]
releaseFunction = "ruleCurve" #·release·function·script
septemberRule = "off" #·september·rule:·"off"·o
limitType = "Bv7_glamUpdates" #·flow·limits·script·name
stlawRouting = "stlaw" #·st.·lawrence·routing·sc
trace = "historic" #·folder·name·of·the·hydr
inputFile = "1961_2020/12month_sqAR" #·path·and·file·name·of·t
[optimizationParameters]
numDV = 9 #·number·of·decision·vari
numObj = 7 #·number·of·objectives
numCon = 0 #·number·of·constraints
nfe = 75000 #·number·of·function·eval
popSize = 100 #·initial·population·size
metFreq = 100 #·frequency·of·function·e
[decisionVariables]
dvName = [#·decision·variables·name
"wetIndicatorThreshold",
"wetConfidenceThreshold",
"C1",
"addC1",
"P1",
"C2",
"P2",
"dryLevelThreshold",
"dryFlowAdjustment",]
lowerBounds = [5500.0, 0.0, 0.0, 0.0, 0.0, 0.5, 0.5, 74.0, -50.0,]
upperBounds = [10000.0, 250.0, 1000.0, 250.0, 2.0, 300.0, 2.0, 76.5, 50.0,]
normalized = "False" #·optimize·normalized·("T
normalizedRange = []
[performanceIndicators]
objectiveFunction = "legacyPIs" #·objective·function·script·name
metricWeighting = "netAnnualAverage" #·weighting·function
objectiveModels = [#·script·names·of·PI·models·to·run·in·`functions/`
"upstreamCoastalImpacts",
"downstreamCoastalImpacts",
"commercialNavigation",
"hydropowerProduction",
"meadowMarsh",
"muskratLodgeViability",
"recreationalBoating"
]
piName = [#·names·of·performance·indicators
"Coastal Impacts: Upstream Buildings Impacted (#)",
dps*
0 0
0
Ln 58, Col 40
Spaces: 4
UTF-8
LF
TOML
no schema selected
Formatting:
functions
Favorites
Macintosh HD
Applications
Documents
Downloads
Desktop
CIGLR
github
loslrRegulation
iCloud
Locations
Google Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
Search
functions
agu2022
Git_Repo
GLRRM
GLRRM-Ontario
lab-manual
loslr_regulation_optimization
plan_2014
Plan-2014-Python
__pycache__
config
dashboard
environment.yml
functions
input
moeaFramework
objectiveFunctions
optimizationSimulation.py
optimizationWrapper.py
output
overview.pptx
README.md
resources
runOptimization_HPC.sh
runOptimization_Local.sh
legacyPIs
README.md
__pycache__
data
functions
objectiveSimulation.py
__pycache__
commercialNavigation.py
downstreamCoastalImpacts.py
hydropowerProduction.py
meadowMarsh.py
muskratLodgeViability.py
recreationalBoating.py
upstreamCoastalImpacts.py

Running the optimization



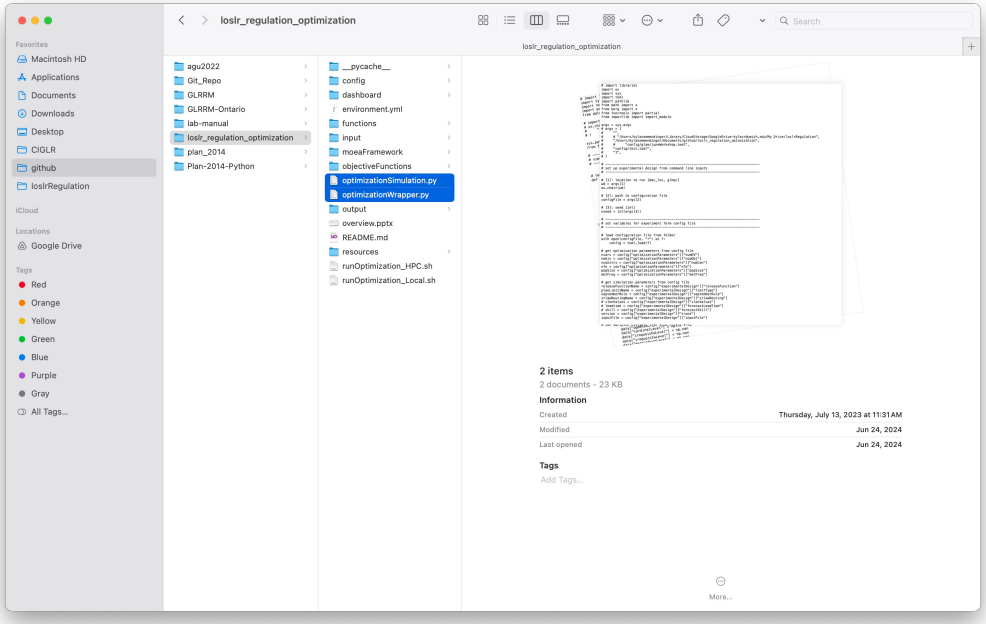
Running the optimization

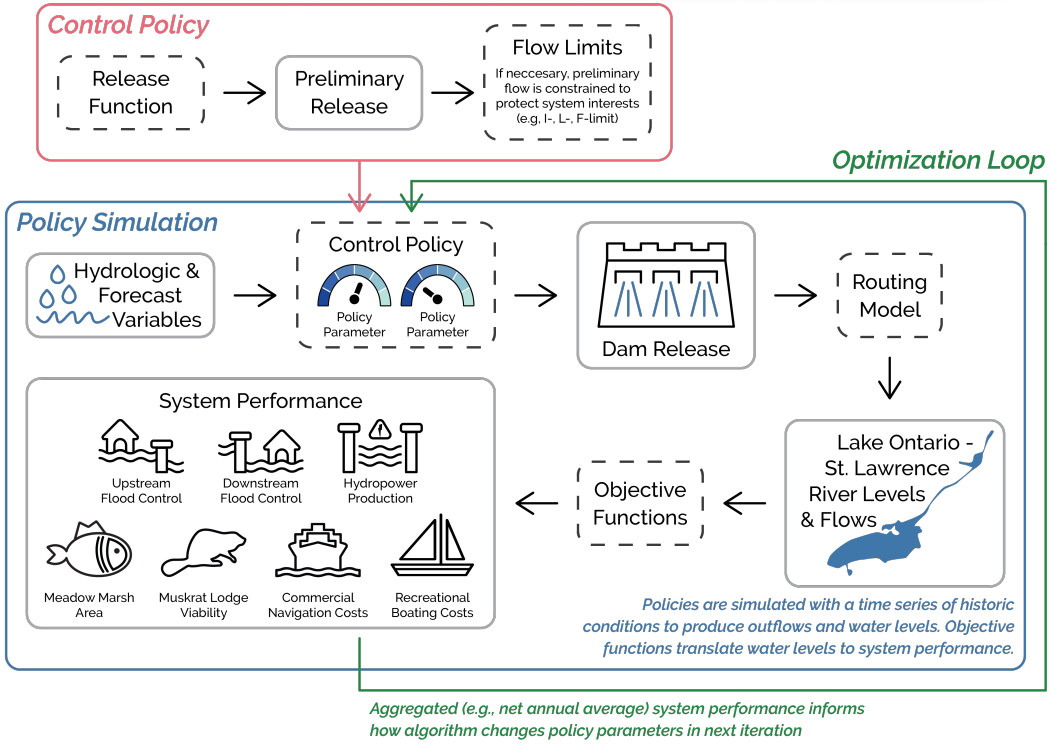



Search
runOptimization_Local.sh
Users > kylasemmendinger > Documents > github > loslr_regulation_optimization > $ runOptimization_Local.sh
#!/bin/bash
loc="$1" #·home·directory·for·the·optimization·code
config="$2" #·path·to·config·file·from·home·directory
nseeds="$3" #·number·of·random·seeds·to·run·optimization
#·array·of·unique·seeds·based·on·number·of·seeds·specified
SEEDS=$(seq 1 ${nseeds})
#·for·loop·to·run·optimizationWrapper.py·for·each·seed
for S in ${SEEDS}
do
python optimizationWrapper.py ${loc} ${config} ${S}
done
#·tells·shell·script
wait <(jobs -p)
dps*
0 0
0
loslr_regulation_optimization — -bash — one_dark — 209×40
~/Documents/github/loslr_regulation_optimization — -bash
kylasemmendinger loslr_regulation_optimization (dps) >> conda activate py-plan2014
(py-plan2014) kylasemmendinger loslr_regulation_optimization (dps) >> ./runOptimization_Local.sh /Users/kylasemmendinger/Documents/github/loslr_regulation_optimization config/glam/juneWorkshop.toml 5

Main scripts

- `optimizationWrapper` sets up the experiment in Borg and calls `optimizationSimulation`
  - Green optimization loop
- `optimizationSimulation` simulates the time series of water levels and system impacts and returns aggregated metrics to Borg
  - Blue simulation loop
- Modules are defined in the `functions/` directory
  - Pink control policy loop (and routing)





Other things to note:

- Syntax might look funky… code has been optimized for speed!
- Have not tested on Windows
  - Pathing might be an issue, just be aware!
- Templates are included to help you write new functions
  - Required outputs are described in each function
- Setting up the many-objective evolutionary algorithm, Borg
  - Documentation is linked in the repo
- Compiling the MOEAFramework (to calculate Borg metrics)
  - Documentation is linked in the repo