NAVAL
POSTGRADUATE
SCHOOL

Leveraging High Performance Computing for Salvo Exchange Simulations

John Berner, Capt, USMC
Michael Galvan, Capt, USMC
Ryan Harris, Capt, USMC
Thomas Stone, USN

Agenda

- Introduction to Salvo Models
- Simulation Concept
- Salvo Models App Workflow
  - Streamlit Interface
  - DOE Modifications
  - Paramiko for HPC Connection
  - Batch Simulations in HPC
  - Large Data Preprocessing
  - Visualizations
- Conclusion

- The salvo model was originally developed by CAPT Wayne Hughes in 1986 to provide insight into modern naval missile engagements, where traditional continuous-fire models were inadequate.

- The model provides a way to compare naval forces by analyzing offensive, defensive, and staying power during naval combat.

- The salvo model analyzes combat as simultaneous volleys instead of continuous fire, which captures the impact of high-damage, high-risk exchanges.

- Salvo models aid in designing ships and strategies by assessing the balance of offensive, defensive, and staying power in naval engagements.

The Basic Model:

$$\Delta B = \frac{\boxed{\alpha A - b_3 B}}{b_1} \qquad \Delta A = \frac{\boxed{\beta B - a_3 A}}{a_1} \qquad (1)$$

Calculates <u>combat work</u> achieved by a single salvo at any time step (ships out of action)

Combat Power: P_a or P_b measured in hits that damage target force.

A = # of units in force A

B = # of units in force B

α = # of well-aimed missiles fired by each A unit

β = # of well-aimed missiles fired by each B unit

a_1 = # of hits by B's missiles needed to put one A out of action

b_1 = # of hits by A's missiles needed to put one B out of action

a_3 = # of well-aimed missiles destroyed by each A

b_3 = # of well-aimed missiles destroyed by each B

ΔA = # of units in force A out of action from B's salvo

ΔB = # of units in force B out of action from A's salvo

Model Insights: Instability

Previous calculation highlights the concept of instability for weak staying power vs strong combat power.

$$\Delta B = \frac{\alpha A - b_3 B}{\boxed{b_1}} \qquad \Delta A = \frac{\beta B - a_3 A}{\boxed{a_1}} \qquad (1)$$

If staying power cannot be easily and affordably added, then instability is only restored through adding more units (A's and B's of affordable attributes)

This type of instability heavily favors circumstances for unanswered strikes and calls for superior scouting for the first effective attack (embellished model).

"Fire effectively first."
– The Great Wayne P. Hughes Jr.

A simple salvo model demonstrating such instability implies limited value of specific and detailed studies **until** we better understand the ***general*** nature of warship characteristics and modern combat.

Focus on combat power alone increases risk of investing too much capability into a single package. Must consider factors of greater numbers and staying power.

The Embellished Model:

$$\Delta B = \frac{\boxed{(\alpha' A - b_3' B) b_4}}{b_1} \qquad \Delta A = \frac{\boxed{(\beta' B - a'_3 A) a_4}}{a_1}$$

where

α' = fighting power in hits of an attacking unit of side A modified for scouting and training deficiencies and the effect of defender B's distraction chaff

β' = fighting power in hits of an attacking unit of side B modified for scouting and training deficiencies and the effect of defender A's distraction chaff

$a_3{}'$ = hits denied to A by defender counterfire of B, degraded for defender alertness and training deficiencies

$b_3{}'$ = hits denied to B by defender counterfire of A, degraded for defender alertness and training deficiencies

a_4 = A's effectiveness in employing seduction chaff to cause otherwise accurate well aimed B missiles to miss after counterfire has failed

b_4 = B's effectiveness in employing seduction chaff to cause otherwise accurate well aimed A missiles to miss after counterfire has failed

Combat Power P_a or P_b are now based on partial offensive and defensive effectiveness

This model incorporates values for combat features that are difficult to capture quantitatively

Embellished Salvo Model

Modified Combat Effectiveness:

$$\alpha' = \sigma_a \tau_a \rho_b \alpha$$

$$b_3' = \delta_b \tau_b b_3$$

$$\beta' = \sigma_b \tau_b \rho_a \beta$$

$$a_3' = \delta_a \tau_a a_3$$

τ is present in both offensive and defensive effectiveness for both sides

These additional factors can only degrade combat effectiveness, not enhance

where

σ = *Scouting effectiveness.* Takes values between 0 and 1 that measure the extent to which striking power is diminished due to less than perfect targeting and distribution of fire against the target force

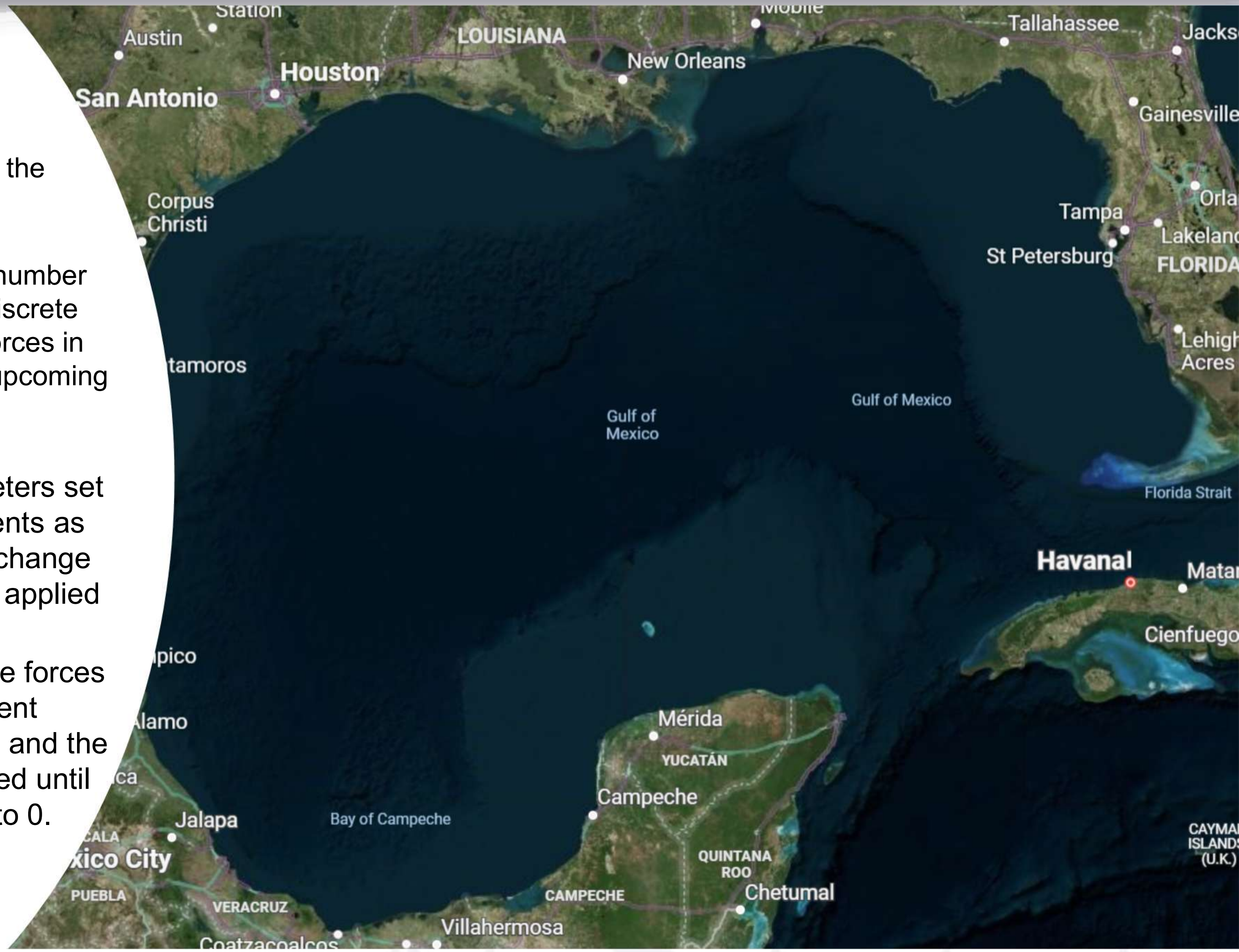
δ = *Defender alertness.* Takes values between 0 and 1 that measure the extent to which counterfire is diminished due to less than perfect readiness or fire control designation to destroy the missiles of an enemy attack

ρ = *Distraction Chaff.* Multiplier between 0 and 1 that reduces the number of accurate shots that must be destroyed by counterfire. Distracts each enemy shot with a fixed probability.

τ = *Skillfulness/Training multiplier.* Multiplier between 0 and 1, the degree to which a firing or target unit fails to achieve its full combat potential due to inadequate training, organization, or motivation.

Simulation Concept

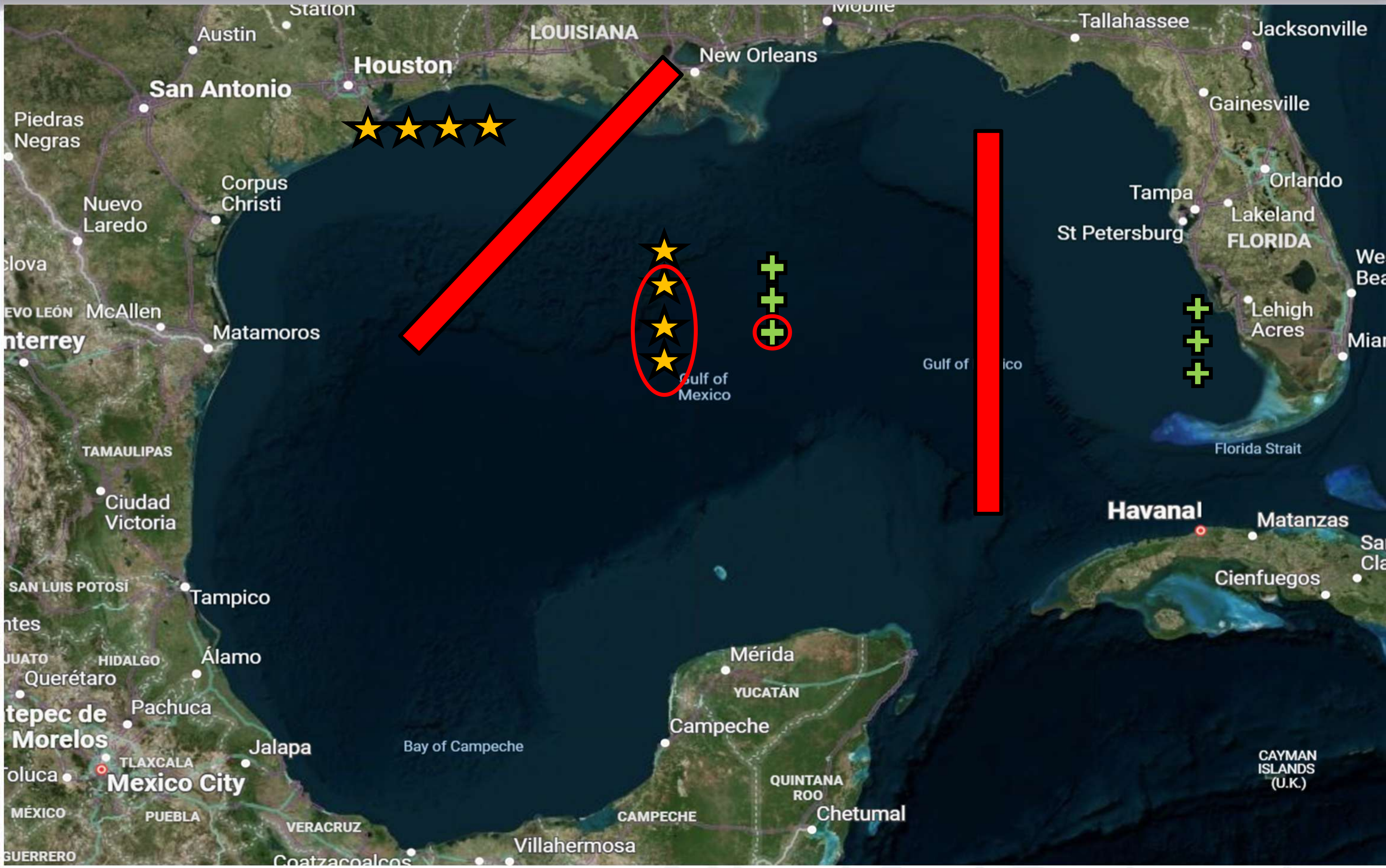
- A_0 and B_0 set at the start of the simulation.
- Half of each side are in the engagement area and the number of forces is selected by a discrete random uniform (1 – # of forces in engagement area) for the upcoming exchange.
- Exchange dynamics are dependent on the parameters set by the design of experiments as the inputs to the salvo exchange equations and losses are applied to the number of forces.
- Reserves are added to the forces available in the engagement area, forces are selected, and the salvo exchange is repeated until one of the sides is equal to 0.



Half of the Total Force is Available for each Engagement

NAVAL POSTGRADUATE SCHOOL

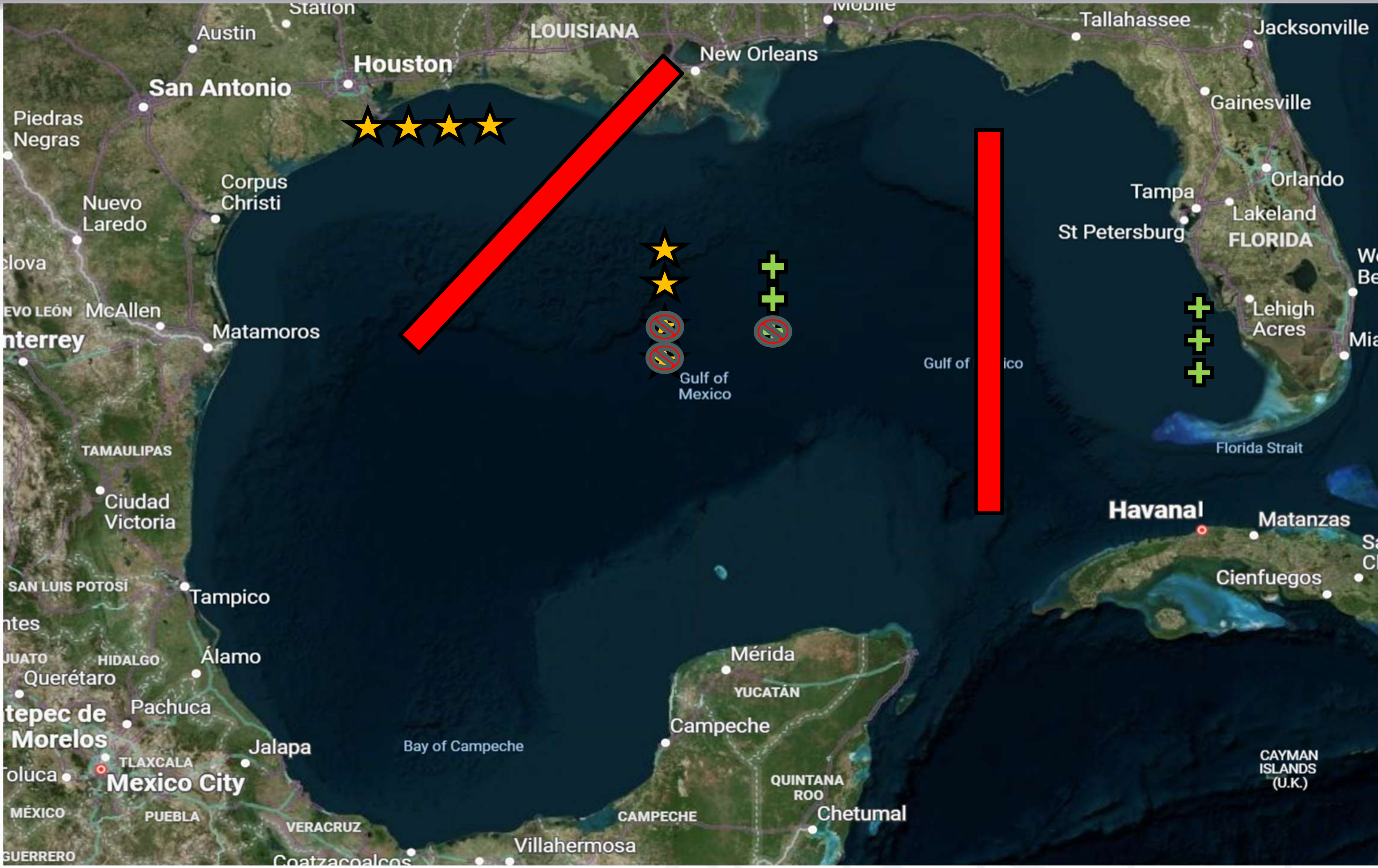
Forces are Randomly Selected for the Exchange



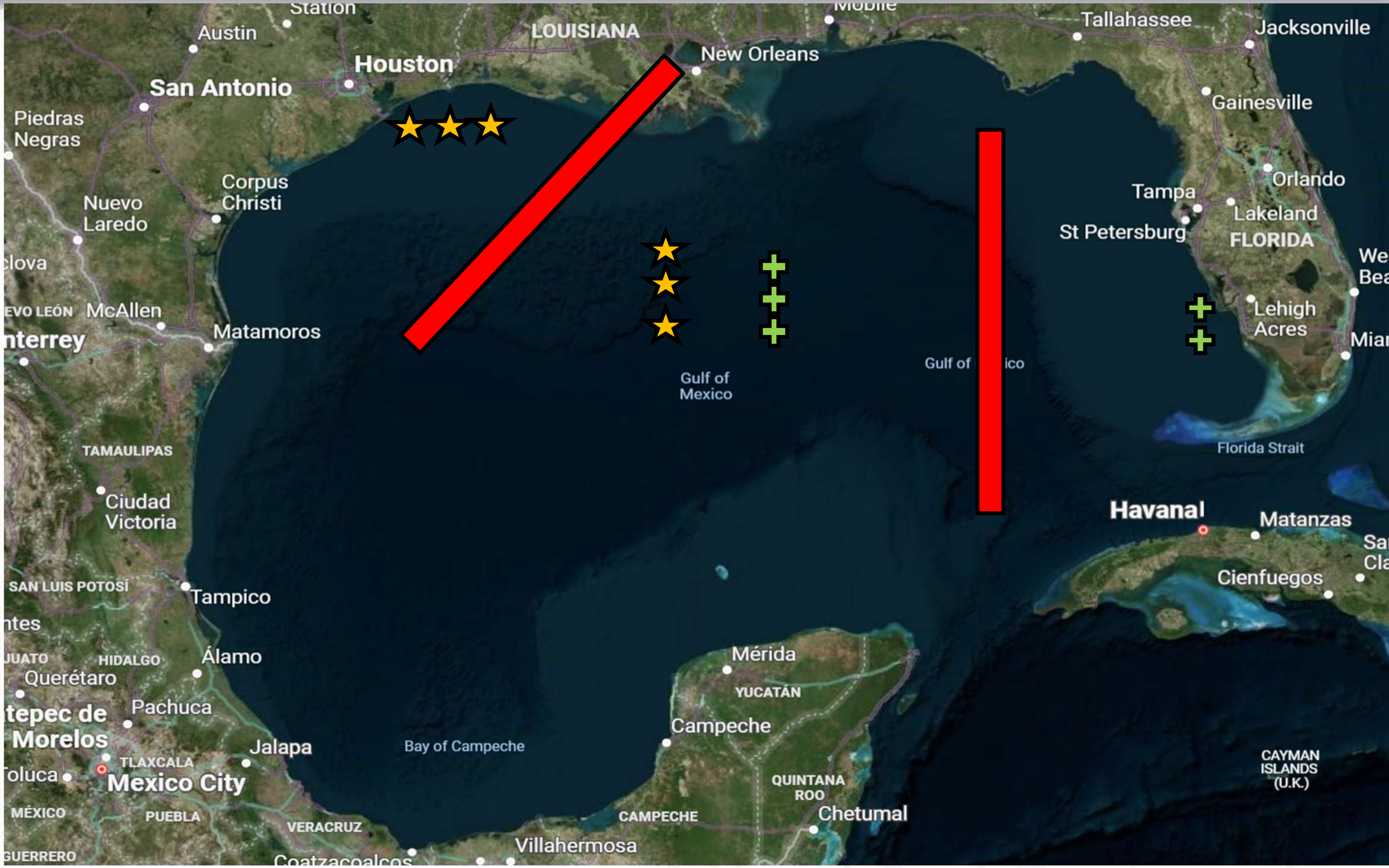
Conduct Salvo Exchange Based on Design Point Parameters

NAVAL
POSTGRADUATE
SCHOOL

Forces are Replaced and Half of the Total Force is Available for Next Exchange



Forces are Replaced and Half of the Total Force is Available for Next Exchange



Parameters (Basic Equation)

Basic Random Selection Model

- **A**: initial number of units in force A
- **B**: initial number of units in force B
- **α** : number of **well aimed** missiles fired by each A unit.
- **β** : number of **well aimed** missiles fired by each B unit.
- **a_1** : number of hits by B's missiles needed to put one A unit out of action.
- **b_1** : number of hits by A's missiles needed to put one B unit out of action.
- **a_3** : number of **well aimed** missiles **destroyed** by each A.
- **b_3** : number of **well aimed** missiles **destroyed** by each B.

Stochastic Basic Model

- **A**: initial number of units in force A
- **B**: initial number of units in force B
- **α** : number of missiles fired by each A unit.
- **β** : number of missiles fired by each B unit.
- **a_1** : number of hits by B's missiles needed to put one A unit out of action.
- **b_1** : number of hits by A's missiles needed to put one B unit out of action.
- **a_3** : number of missiles that can be destroyed by each A.
- **b_3** : number of missiles that can be destroyed by each B.
- **$P(A_{\text{Well-Aimed Missile}})$** : probability that an A unit fires a missile that can hit a B unit
- **$P(B_{\text{Well-Aimed Missile}})$** : probability that a B unit fires a missile that can hit an A unit
- **$P(A_{\text{Destroy Well-Aimed Missile}})$** : probability that an A unit can destroy a missile that can hit an A unit
- **$P(B_{\text{Destroy Well-Aimed Missile}})$** : probability that a B unit can destroy a missile that can hit a B unit

Parameters (Modified Equation)

Modified Random Selection Model

- **A**: initial number of units in force A
- **B**: initial number of units in force B
- **α** : number of **well aimed** missiles fired by each A unit.
- **β** : number of **well aimed** missiles fired by each B unit.
- a_1 : number of hits by B's missiles needed to put one A unit out of action.
- b_1 : number of hits by A's missiles needed to put one B unit out of action.
- a_3 : **number** of **well aimed** missiles **destroyed** by each A.
- b_3 : **number** of **well aimed** missiles **destroyed** by each B.
- a_4 : probability that accurate shots miss an A unit after their counterfire has failed.
- b_4 : probability that accurate shots miss a B unit after their counterfire has failed.
- σ_A : scouting effectiveness due to less than perfect targeting of A units.
- σ_B: scouting effectiveness due to less than perfect targeting of B units.
- τ_A : training effectiveness of A units.
- τ_B : training effectiveness of B units.
- δ_A : alertness of unit A's defense that is degraded by less than perfect readiness or fire control.
- δ_B : alertness of unit B's defense that is degraded by less than perfect readiness or fire control.
- ρ_A : effectiveness of unit A chaff that draws off shots before its counterfire against B's missiles.
- ρ_B : effectiveness of unit B chaff that draws off shots before its counterfire against A's missiles.

Stochastic Modified Model

- **A**: initial number of units in force A
- **B**: initial number of units in force B
- **α** : number of missiles fired by each A unit.
- **β** : number of missiles fired by each B unit.
- a_1 : number of hits by B's missiles needed to put one A unit out of action.
- b_1 : number of hits by A's missiles needed to put one B unit out of action.
- a_3 : number of missiles that can be destroyed by each A.
- b_3 : number of missiles that can be destroyed by each B.
- a_4 : probability that accurate shots miss an A unit after their counterfire has failed.
- b_4 : probability that accurate shots miss a B unit after their counterfire has failed.
- σ_A : scouting effectiveness due to less than perfect targeting of A units.
- σ_B: scouting effectiveness due to less than perfect targeting of B units.
- τ_A : training effectiveness of A units.
- τ_B : training effectiveness of B units.
- δ_A : alertness of unit A's defense that is degraded by less than perfect readiness or fire control.
- δ_B : alertness of unit B's defense that is degraded by less than perfect readiness or fire control.
- ρ_A : effectiveness of unit A chaff that draws off shots before its counterfire against B's missiles.
- ρ_B : effectiveness of unit B chaff that draws off shots before its counterfire against A's missiles.
- $P(A_{\text{Well-Aimed Missile}})$: probability that an A unit fires a missile that can hit a B unit
- $P(B_{\text{Well-Aimed Missile}})$: probability that a B unit fires a missile that can hit an A unit
- $P(A_{\text{Destroy Well-Aimed Missile}})$: probability that an A unit can destroy a missile that can hit an A unit
- $P(B_{\text{Destroy Well-Aimed Missile}})$: probability that a B unit can destroy a missile that can hit a B unit

App Work Flow

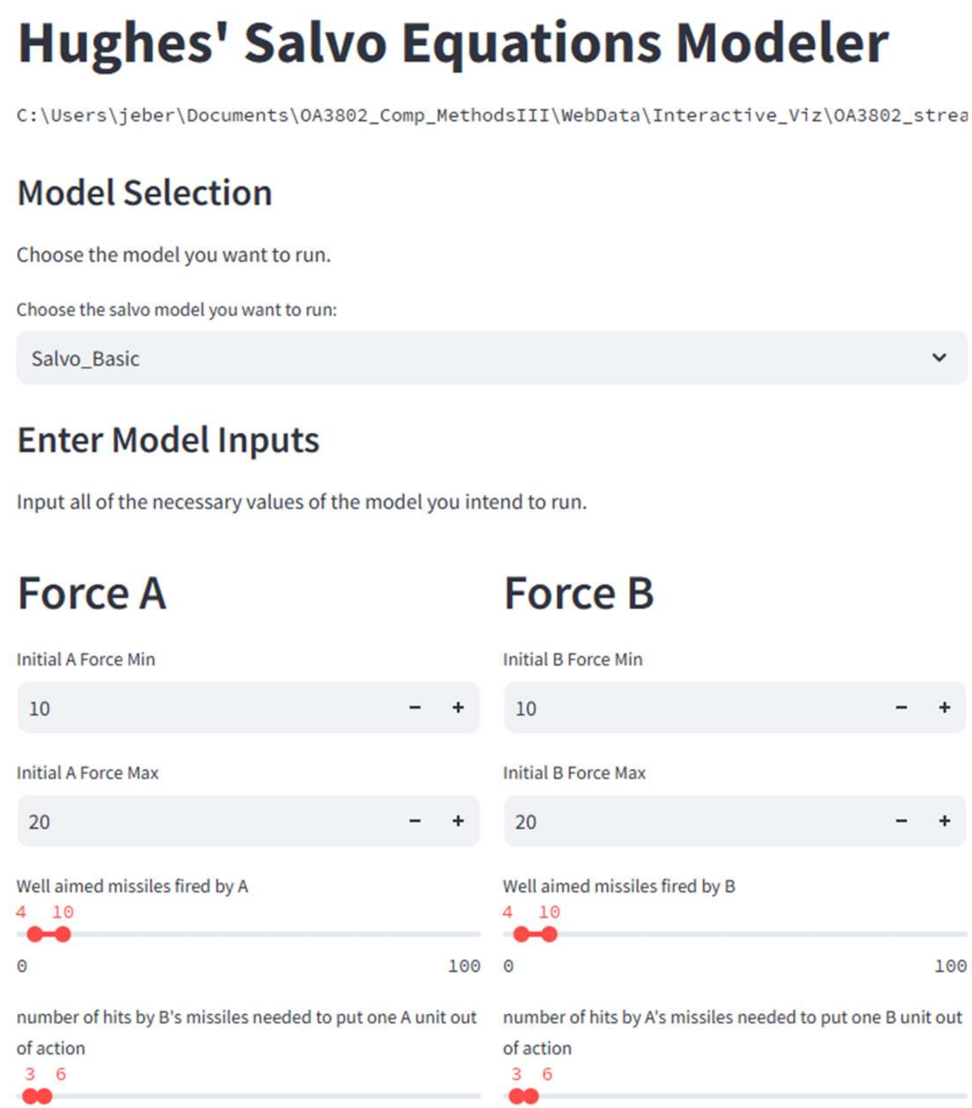
plotly

Streamlit

Paramiko establishes a secure shell to NPS Hamming, allowing bash commands to run transparently in the background of the streamlit app, and provide the user updates

User Inputs from Streamlit

- User selects the model
  - Basic, Basic Stochastic, Modified, Modified Stochastic
- User selects minimum and maximum values that they are interested in studying
- Slide bars were used for user functionality
- A_0 and B_0 range of values were too large for the user to control the values selection



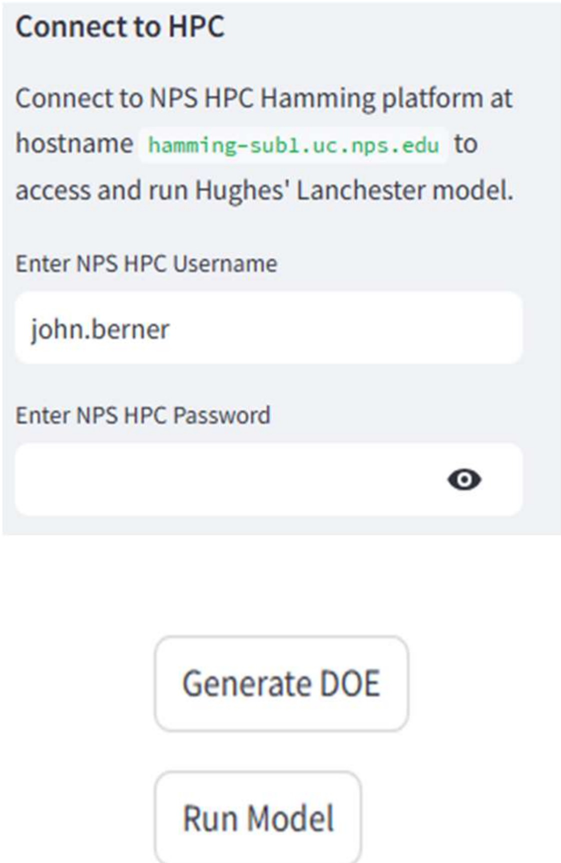
Streamlit App

- User requires an account for NPS HPC
- App is designed to be run with no coding necessary once it is launched
- App must be launched from a terminal.
- This is important for running the model seamlessly since file management can get complicated.
- Absolute Paths with modular variables.

```
(base) C:\Users\jeber\Documents\OA3802_Comp_MethodsIII\WebData\Interactive_Viz\OA3802_streamlit_exercise_20241021153925112\OA3802_streamlit_exercise\OA3802_streamlit_SSH_PE\Salvo\Salvo>streamlit run Hughes_Salvo_App_V2.py
```

```
f"{present_dir}/{model}_HPC.py", f"/home/{username}/Salvo_outputs/{model}_HPC.py")
```

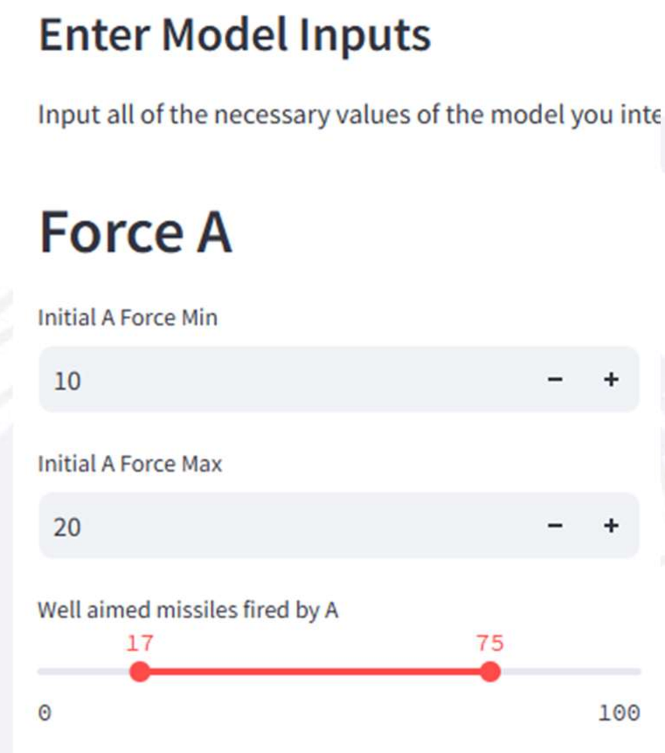
User Interactions



User logs in with their own NPS HPC account info. Username is stored for use in bash commands

User then inputs all required variables for their selected model using input boxes and sliding bars.

Code assumes user can run Python on HPC already



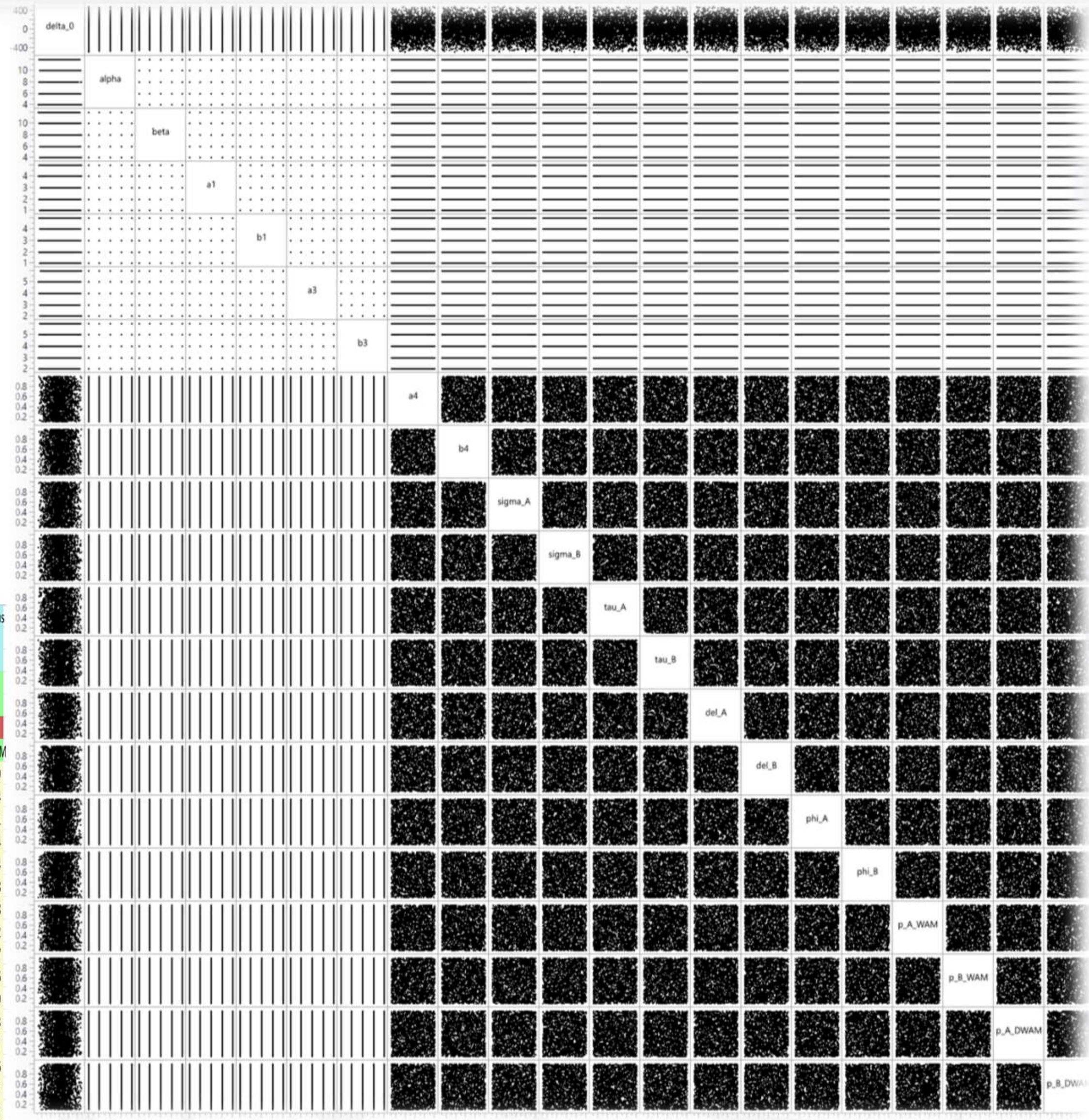

```python
if st.button('Run Model'):
    if connected:
        prep_model(model, client, username)
        run_model(present_dir, model, client, username)
        run_robust(present_dir, model, client, username)
        clear_outputs(present_dir, model, client, username)

    else:
        st.write('Connect to HPC and try again.')
```

```python
if st.button('Generate DOE'):
    if connected:
        generate_DOE(A_val, alpha, a1_val, a3_val,
    else:
        st.write('Connect to HPC and try again.')
```

User Inputs for Design of Experiments

- (8) Discrete Variables
  - A_0, B_0, α, β, a_1, b_1, a_3, b_3
- (14) Continuous Variables
  - a_4, b_4, σ_A, σ_B, τ_A, τ_B, δ_A, δ_B, ρ_A, ρ_B, $P(A_{\text{Well-Aimed Missile}})$, $P(B_{\text{Well-Aimed Missile}})$, $P(A_{\text{Destroy Well-Aimed Missile}})$, $P(B_{\text{Destroy Well-Aimed Missile}})$
- Using Python library "xlwings", excel was appended to reflect the range that the user is interested in
- CSV is created for the (2000) design points that the simulations will reference for the initial parameters

discrete	discrete	discrete	discrete	discrete	discrete	discrete	discrete	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous	continuous
5	5	5	5	5	5	100	100	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000	2000
MI	MI	MI	MI	MI	MI	MQI	MQI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI	MI
2	2	1	1	2	2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
6	6	5	5	6	6	100	100	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	2	1	1	1	1	5	5	6	6	6	6	6	6	6	6	6	6	6	6	6	6
alpha	beta	a1	b1	a3	b3	A_0	B_0	a4	b4	sigma_A	sigma_B	tau_A	tau_B	del_A	del_B	phi_A	phi_B	p_A_WAM	p_B_WAM	A_DWAM	B_DWAM
10	10	5	5	3	2	235	65	0.312956	0.589845	0.95903	0.938769	0.73932	0.727614	0.68034	0.245873	0.472786	0.510155	0.144572	0.258929	0.210305	0.258929
8	8	1	1	2	5	105	185	0.572736	0.754627	0.925713	0.553827	0.918509	0.928864	0.848724	0.989195	0.172036	0.971186	0.283242	0.925713	0.649725	0.688444
4	8	1	2	4	5	350	445	0.687994	0.984242	0.3008	0.268384	0.353027	0.341321	0.54032	0.333217	0.388594	0.864932	0.535368	0.964432	0.430465	0.381391
12	8	1	5	5	3	30	160	0.112606	0.221561	0.698349	0.878889	0.291796	0.513307	0.13962	0.196348	0.206703	0.654677	0.182391	0.545723	0.646123	0.767234
4	10	4	5	4	5	195	500	0.257579	0.893747	0.127914	0.69925	0.974787	0.674037	0.301701	0.658729	0.362931	0.671786	0.73977	0.366083	0.270185	0.387694
12	10	3	1	4	6	345	150	0.958579	0.136018	0.281441	0.481791	0.676288	0.35933	0.827114	0.101801	0.937419	0.524112	0.857279	0.933367	0.71906	0.356628
4	8	2	4	2	5	50	250	0.662331	0.457479	0.357529	0.722661	0.215258	0.777589	0.915808	0.724462	0.929765	0.415158	0.823962	0.36068	0.483592	0.516458
4	4	2	3	5	5	385	135	0.483142	0.313407	0.757779	0.567784	0.428214	0.97974	0.470535	0.322861	0.618209	0.227414	0.66098	0.11981	0.985143	0.764532
10	4	5	1	3	6	80	205	0.321511	0.647924	0.534917	0.462881	0.988294	0.994147	0.614607	0.458379	0.158979	0.475938	0.222011	0.225613	0.736168	0.213007
8	10	4	2	3	4	365	330	0.809105	0.234167	0.551126	0.977939	0.337719	0.437219	0.225613	0.815858	0.223362	0.486743	0.342671	0.421011	0.362031	0.912206
10	10	5	3	5	3	75	390	0.686643	0.95903	0.613257	0.545723	0.493047	0.992796	0.918959	0.914907	0.786143	0.381391	0.330065	0.710055	0.903652	0.477739
4	4	4	4	3	2	140	490	0.253077	0.255328	0.682141	0.886993	0.202651	0.658729	0.165283	0.552926	0.402101	0.941471	0.484042	0.996398	0.271536	0.766783
12	12	3	1	3	6	160	475	0.835218	0.590295	0.882941	0.832066	0.136018	0.187794	0.297199	0.817659	0.769035	0.528164	0.241821	0.552926	0.783892	0.53987
8	12	2	4	6	4	395	65	0.987844	0.269285	0.998199	0.507454	0.806403	0.594797	0.548424	0.424612	0.753727	0.909505	0.526363	0.772636	0.447574	0.271536
10	6	4	2	5	6	350	145	0.846473	0.106753	0.495748	0.307554	0.2999	0.290445	0.24002	0.485843	0.671336	0.489895	0.701501	0.206253	0.44037	0.44037
6	12	3	5	3	5	295	385	0.937869	0.852776	0.267034	0.790645	0.653777	0.378689	0.413807	0.470535	0.796948	0.126563	0.154927	0.897349	0.473687	0.38049
4	4	4	4	3	3	190	30	0.249025	0.376438	0.588494	0.675838	0.52051	0.309805	0.954077	0.275138	0.970735	0.124762	0.527264	0.23912	0.134667	0.293147
12	12	3	5	5	5	155	245	0.576338	0.341771	0.217959	0.303952	0.656478	0.153127	0.198149	0.781641	0.649725	0.772186	0.996848	0.484042	0.671336	0.453877
12	8	1	2	3	4	230	465	0.531766	0.589395	0.907704	0.168884	0.145923	0.138719	0.970285	0.294047	0.987844	0.643422	0.193647	0.334117	0.623162	0.684842
12	8	3	5	6	5	445	230	0.905903	0.607854	0.333217	0.475038	0.641621	0.597499	0.573187	0.968484	0.476388	0.977039	0.107654	0.951826	0.82081	0.355278
10	8	5	3	4	5	95	180	0.511506	0.745173	0.550225	0.72086	0.36023	0.383642	0.917609	0.854127	0.409305	0.596148	0.775788	0.866283	0.127014	0.372836
10	10	3	3	6	4	220	125	0.391746	0.223812	0.366083	0.710955	0.312506	0.944622	0.903652	0.890145	0.841071	0.95948	0.635768	0.291346	0.428214	0.570935
8	12	3	5	3	6	90	10	0.761381	0.642971	0.863582	0.430915	0.921661	0.28009	0.771286	0.721761	0.941021	0.133317	0.130165	0.408404	0.738869	0.445773



NAVAL POSTGRADUATE SCHOOL

High Performance Computer (HPC)

- Each batch job will execute a .py file for the corresponding model that will run (2000) simulations
- The batch job will run multiple iterations of the .py file and the random seed is set based on the iteration of the batch job.
- The number of units lost by each force and the winner of each observation of each design point is recorded in a .json file and saved under a data outputs directory
- We executed 500 iterations for each design point and the number of simulations would take an extensive amount of time to complete without HPC
  - The memory required for each model's outputs was estimated to take 4.5 GB


```json
{"remaining_A": 208,
"remaining_B": 0,
"delta_A": 27,
"delta_B": 65,
"total_wins_A": 8,
"total_wins_B": 2,
"total_exchanges": 10,
"winner_A": 1,
"winner_B": 0,
"tie": 0}
```


```sh
#!/bin/sh
#SBATCH --job-name=Basic
#SBATCH --array=1-500
#SBATCH -N 5
#SBATCH --mem-per-cpu=2048M
#SBATCH --cpus-per-task=5
#SBATCH --output=print_outputs/Basic/DP%a.txt

. /etc/profile
module load lang/python
source /smallwork/$USER/comp3/bin/activate

python Salvo_Basic_HPC.py ${SLURM_ARRAY_TASK_ID}

```

High Performance Computer (HPC) Continued

- The data from each simulation needed to be processed to calculate the means and standard deviations for the units lost by each force, and the proportion of wins for either side.
- Due to the size of the folder, the memory that was required to execute these calculations was too large for the user’s local machine

```
for filename in os.listdir('./data_output/Salvo_Basic/'):
    if filename.endswith(".json"):
        DP = int(filename.split('_')[0][2:])
        obs_number = int(filename.split('_')[1].split('.')[0][4:])

        with open(os.path.join('./data_output/Salvo_Basic/', filename), 'r') as f:
            json_data = json.load(f)
        try:
            data_dict[DP]['remaining_A'].append(json_data['remaining_A'])
            data_dict[DP]['remaining_B'].append(json_data['remaining_B'])
            data_dict[DP]['delta_A'].append(json_data['delta_A'])
            data_dict[DP]['delta_B'].append(json_data['delta_B'])
            data_dict[DP]['total_wins_A'].append(json_data['total_wins_A'])
            data_dict[DP]['total_wins_B'].append(json_data['total_wins_B'])
            data_dict[DP]['total_exchanges'].append(json_data['total_exchanges'])
            data_dict[DP]['winner_A'].append(json_data['winner_A'])
            data_dict[DP]['winner_B'].append(json_data['winner_B'])
            data_dict[DP]['tie'].append(json_data['tie'])

        except:
            data_dict[DP]= {
                'remaining_A': [json_data['remaining_A']],
                'remaining_B': [json_data['remaining_B']],
                'delta_A': [json_data['delta_A']],
                'delta_B': [json_data['delta_B']],
                'total_wins_A': [json_data['total_wins_A']],
                'total_wins_B': [json_data['total_wins_B']],
                'total_exchanges': [json_data['total_exchanges']],
                'winner_A': [json_data['winner_A']],
                'winner_B': [json_data['winner_B']],
                'tie': [json_data['tie']]
                }
```

```
for DP in data_dict.keys():
    data = {
        'index': DP,
        'avg_remaining_A': statistics.mean(data_dict[DP]['remaining_A']),
        'sd_remaining_A': statistics.stdev(data_dict[DP]['remaining_A']),
        'avg_remaining_B': statistics.mean(data_dict[DP]['remaining_B']),
        'sd_remaining_B': statistics.stdev(data_dict[DP]['remaining_B']),
        'avg_delta_A': statistics.mean(data_dict[DP]['delta_A']),
        'sd_delta_A': statistics.stdev(data_dict[DP]['delta_A']),
        'avg_delta_B': statistics.mean(data_dict[DP]['delta_B']),
        'sd_delta_B': statistics.stdev(data_dict[DP]['delta_B']),
        'FER': FER_across_DP(data_dict, DP),
        'avg_total_wins_A': statistics.mean(data_dict[DP]['total_wins_A']),
        'sd_total_wins_A': statistics.stdev(data_dict[DP]['total_wins_A']),
        'avg_total_wins_B': statistics.mean(data_dict[DP]['total_wins_B']),
        'sd_total_wins_B': statistics.stdev(data_dict[DP]['total_wins_B']),
        'avg_total_exchanges': statistics.mean(data_dict[DP]['total_exchanges']),
        'sd_total_exchanges': statistics.stdev(data_dict[DP]['total_exchanges']),
        'prop_A_wins_exchanges': sum(data_dict[DP]['total_wins_A'])/sum(data_dict[DP]['total_exchanges']),
        'prop_B_wins_exchanges': sum(data_dict[DP]['total_wins_B'])/sum(data_dict[DP]['total_exchanges']),
        'prop_A_wins_overall': sum(data_dict[DP]['winner_A'])/len(data_dict[DP]['winner_A']),
        'prop_B_wins_overall': sum(data_dict[DP]['winner_B'])/len(data_dict[DP]['winner_B']),
        'prop_tie_overall': sum(data_dict[DP]['tie'])/len(data_dict[DP]['tie'])
    }
    outputs.append(data)

df_DOE = pd.read_csv('sim_DOE.csv')
df_DOE.index += 1
df = pd.DataFrame(outputs)
merged_df = df_DOE.merge(df, left_index=True, right_on='index', how='left')

merged_df.to_csv('sim_DOE_results_Salvo_Basic.csv', header=True, index=False)
```

Results

Visualizations

Streamlit app hosts four variable heatmaps with dropdown selectors:

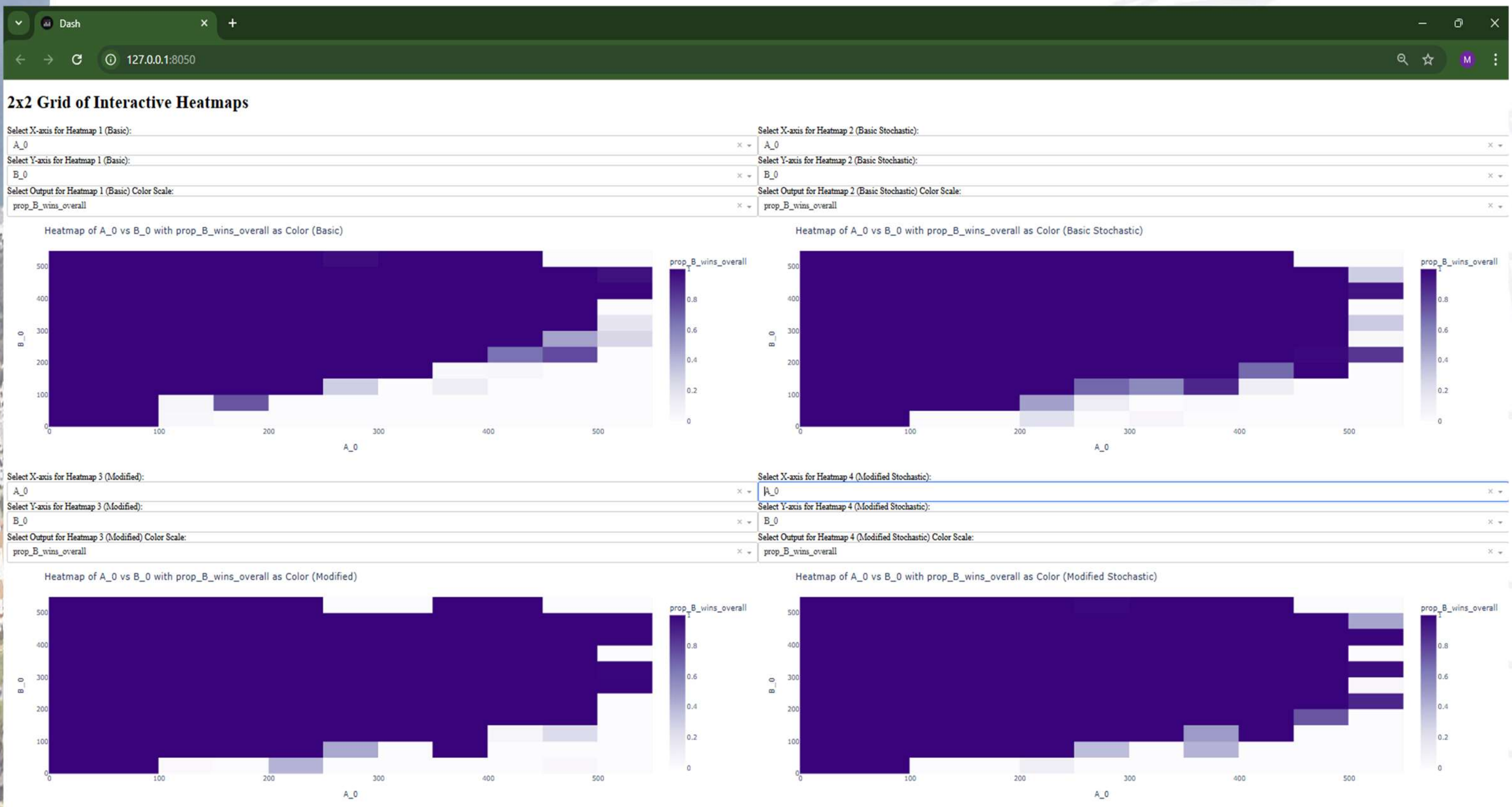
- x and y (Input parameters)
- Output (specific simulation result) dictates color

Here we have:

x: A_0 (initial A forces)

y: B_0 (initial B forces)

Output: Proportion B wins overall



Conclusions

Comp 3 Takeaways:

- Not all HPC tunnels are created equal
  - MobaXTerm vs. Paramiko vs. Other solutions
- Dashboards are helpful, but too much going on in the background can get complicated
- Output management: SQL vs. Other data storage methods (pandas, .csv, .json)
  - Don't reinvent the wheel
- File paths and file permissions are easy ways for code to break
- Scouting is the key to success and survival. This applies to delivering weapons at range and disrupting the enemies sensing and C2
- Maximum fighting strength (regardless of how it is achieved) is the proper warship design goal. Further research is needed comparing the relative worth of offensive and defensive power with the number of warships and staying power

Future Work:

- Build out the model repository with other Salvo equations
- Further develop “immediate analysis” aids (think mini-STORMMiner) in Dash

Questions?