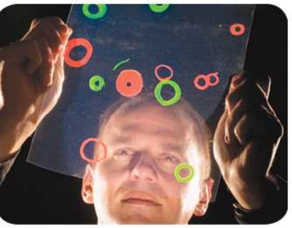
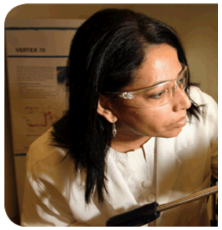
SAND2017-10615PE





A Framework for Optimal Sensor Placement Built on Pyomo

Bethany Nicholson

Katherine Klise

Carl Laird

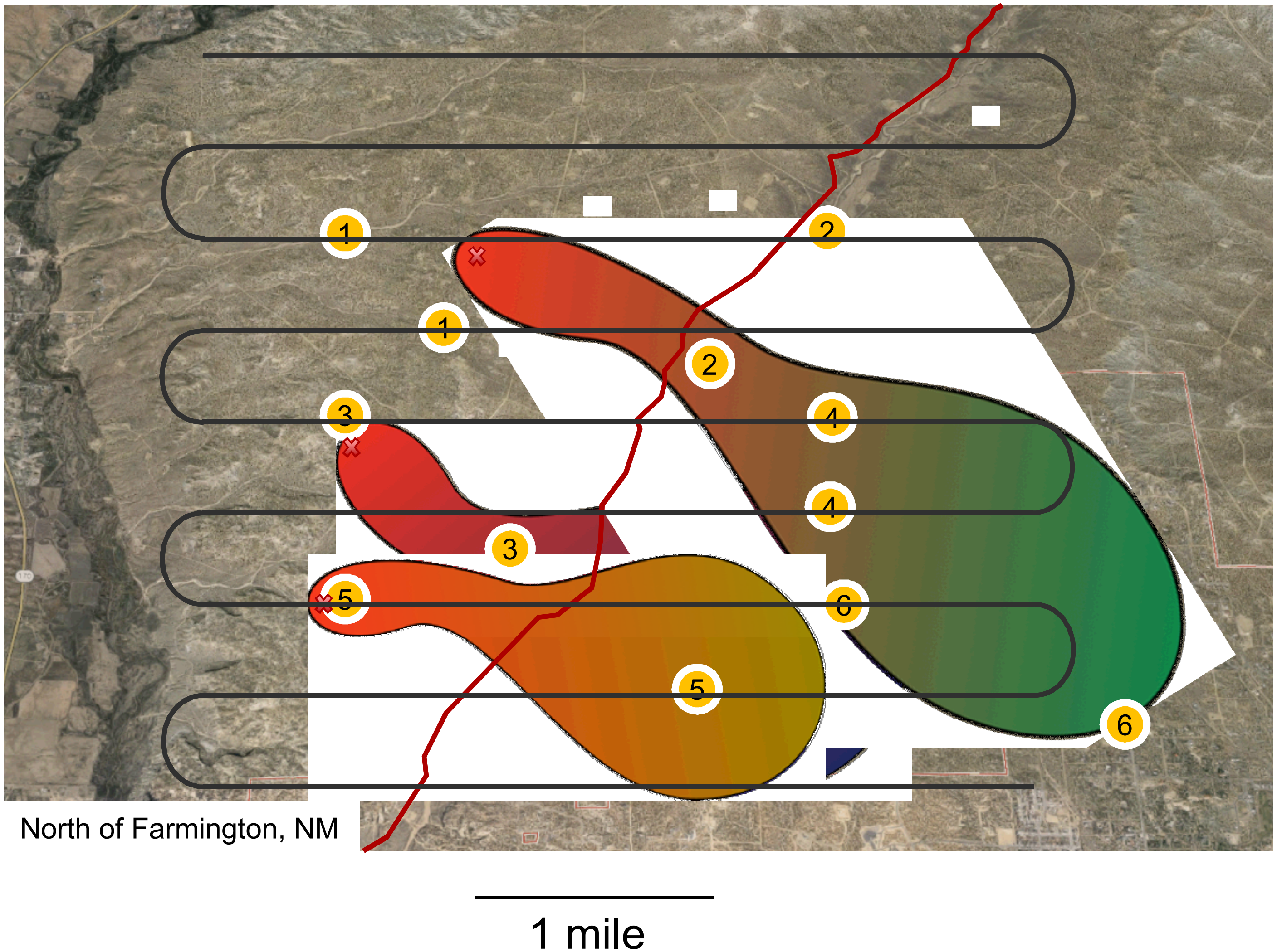
Sandia National Laboratories

Albuquerque, NM

PyomoFest Trondheim October 3 – 5, 2017

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.

Motivating Application: Detecting Gas Emissions

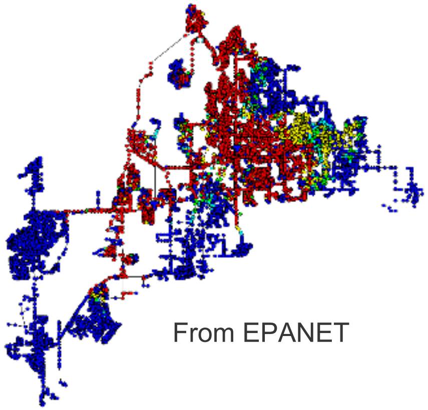


Challenges

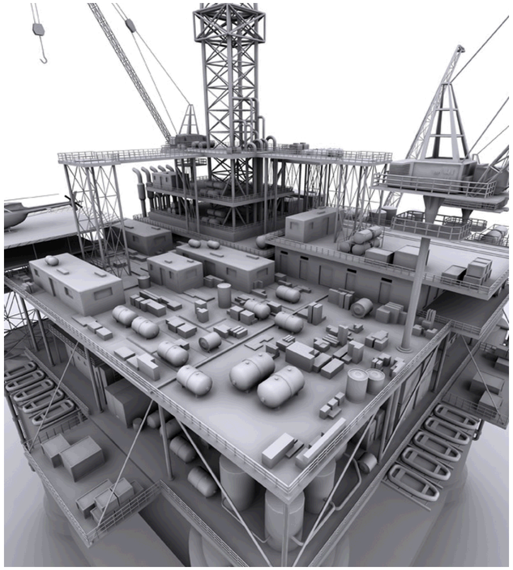
- Different types of monitoring strategies
  - Where should sensors be placed and how should they be operated to…
    - Detect abnormal leaks quickly?
    - Provide constant monitoring?
    - Identify the leak locations?
    - Quantify emissions?
- Tradeoff between sensor cost and detection capability
  - Is it better to use numerous cheap detectors or use a single expensive detector?
  - What sensor attributes are most important for detection?
  - Should sensors be fixed or mobile?
- Emissions are highly variable
  - Rare super emitter and pervasive small leaks
  - Transport governed by complex atmospheric conditions
- Need to incorporate uncertainty in:
  - Leak location
  - Weather, wind direction, wind speed

Other Applications

- Water security
- Monitoring seismic activity
- Fire detection in buildings
- Gas detection at industrial facilities
- Placing surveillance cameras
- etc.



From EPANET



Marsh (2012). The 100 largest losses 1972-2011. London, United Kingdom.

Goal

Develop methods and software to determine optimal **sensor placement** and **sensor technology** to improve the effectiveness of monitoring strategies

Optimization Formulation

$$\min \quad \sum_{a \in A} \alpha_a \sum_{i \in \mathcal{L}_a} D_{a,i} x_{a,i}$$

Minimizes the expected impact across all scenarios

s.t.

$$s_l \in \{0,1\} \qquad \forall\, l \in L$$

Binary variable reflecting existence of a sensor

$$0 \leq x_{a,i} \leq 1 \qquad \forall\, a \in A,\ i \in \mathcal{L}_a$$

Continuous variable representing the “first to detect”

$$\sum_{l \in L} s_l \leq p$$

Constraint limiting the number of sensors allowed

$$\sum_{i \in \mathcal{L}_a} x_{a,i} = 1 \qquad \forall\, a \in A$$

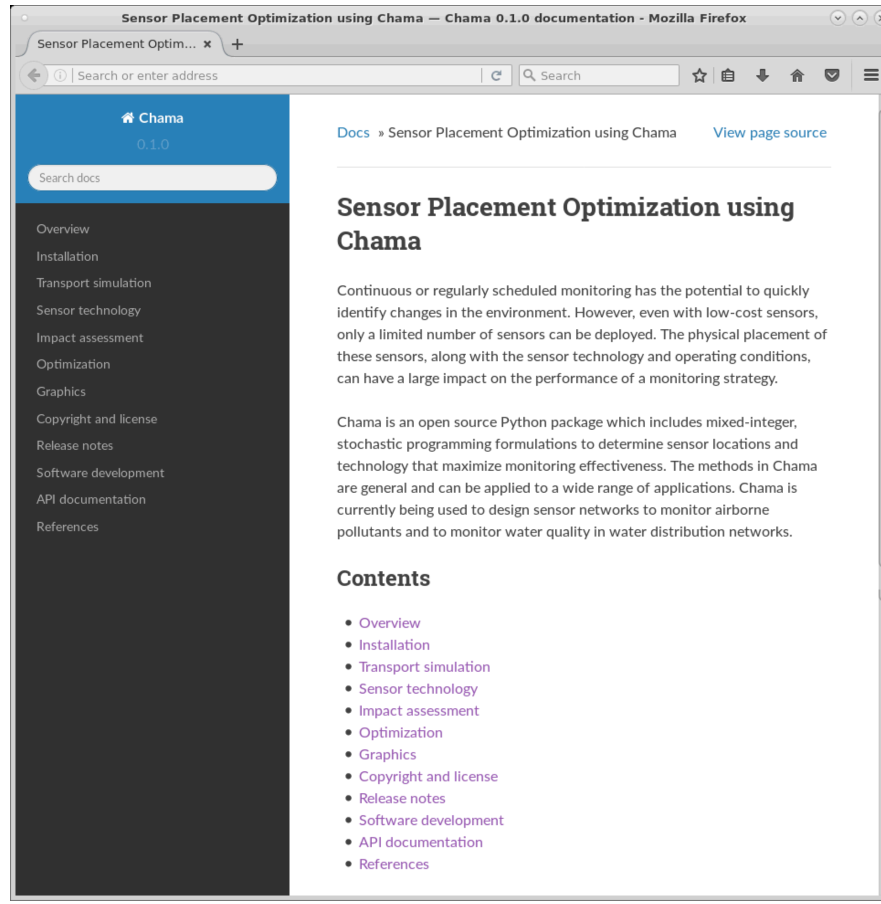
Constraint forcing one sensor location to be the “first to detect”

$$x_{a,i} \leq s_i \qquad \forall\, a \in A,\ i \in \mathcal{L}_a$$

A sensor location can only claim detection if a sensor exists in that location

Software: Chama

- Series of extensible modules
  - Additional dispersion models, sensor types, and optimization formulations could be included
- User can enter the workflow at any stage/module
- Leverages Sandia developed Pyomo software, http://www.pyomo.org/
- First release in October 2017
- Uses Numpy, Pandas, Scipy, Matplotlib

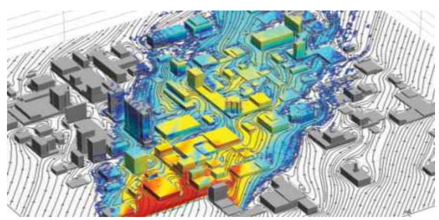


Sensor Placement Framework

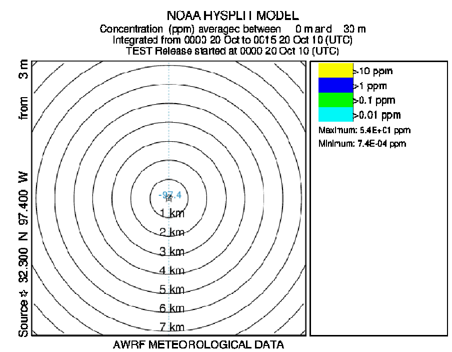
Transport Simulation → Sensor Technology → Impact Assessment → Optimization → Graphics

```
>>> print(signal)
   X  Y  Z   T    S1    S2    S3
0  1  1  1   0  0.00  0.00  0.00
1  1  1  1  10  0.00  0.00  0.01
2  1  1  1  20  0.00  0.00  0.00
3  2  1  1   0  0.20  0.20  0.20
4  2  1  1  10  0.32  0.14  0.14
5  2  1  1  20  0.45  0.58  0.58
```

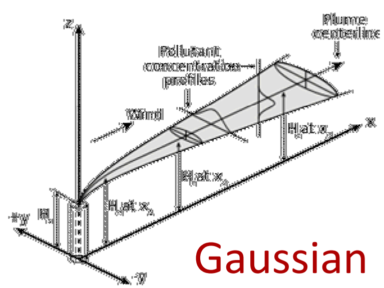
- Need a set of precomputed transport simulations (scenarios) to generate a **signal** under different conditions
- Scenarios should capture uncertainty in weather conditions, infrastructure, emission rate, etc.
- Scenario signals can be generated:
  - Externally:



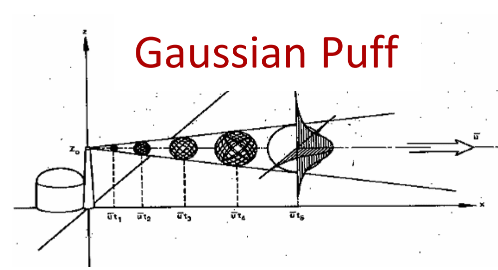
Fast Building-Aware Atmospheric Dispersion Modeling
http://www.lanl.gov/projects/quic/



  - Internally:



Gaussian Plume

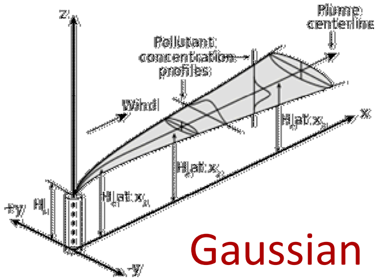


Gaussian Puff

Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → Optimization → Graphics

Ex) Running internal Gaussian Plume transport simulation



Gaussian Plume

Define the simulation grid

```
>>> x_grid = np.linspace(-100, 100, 21)
>>> y_grid = np.linspace(-100, 100, 21)
>>> z_grid = np.linspace(0, 40, 21)
>>> grid = chama.transport.Grid(x_grid, y_grid, z_grid)
```

Define the source (leak)

```
>>> source = chama.transport.Source(-20, 20, 1, 1.5)
```

Define the atmospheric conditions

```
>>> atm = pd.DataFrame({'Wind Direction': [45, 60],
...                     'Wind Speed': [1.2, 1],
...                     'Stability Class': ['A', 'A']}, index=[0, 10])
```

Initialize and run the Gaussian Plume model

```
>>> gauss_plume = chama.transport.GaussianPlume(grid, source, atm)
>>> gauss_plume.run()
>>> signal = gauss_plume.conc
>>> print(signal.head(5))
       X      Y    Z  T    S
0 -100.0 -100.0  0.0  0  0.0
1 -100.0 -100.0  2.0  0  0.0
2 -100.0 -100.0  4.0  0  0.0
3 -100.0 -100.0  6.0  0  0.0
4 -100.0 -100.0  8.0  0  0.0
```

CCR Center for Computing Research

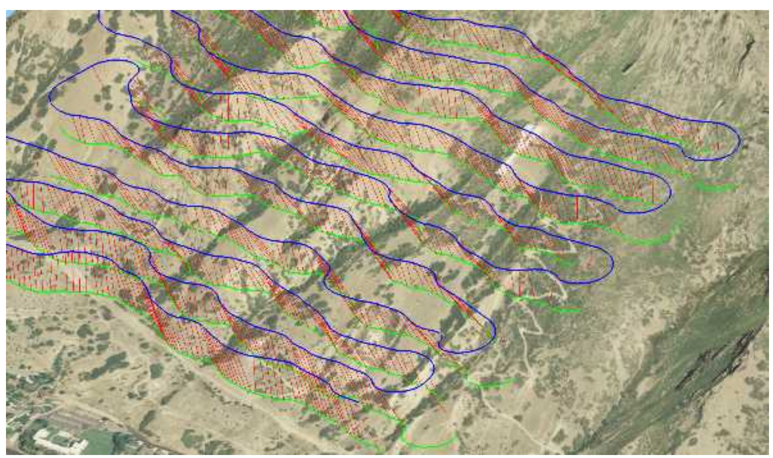
Sandia National Laboratories

Sensor Placement Framework

Sandia National Laboratories

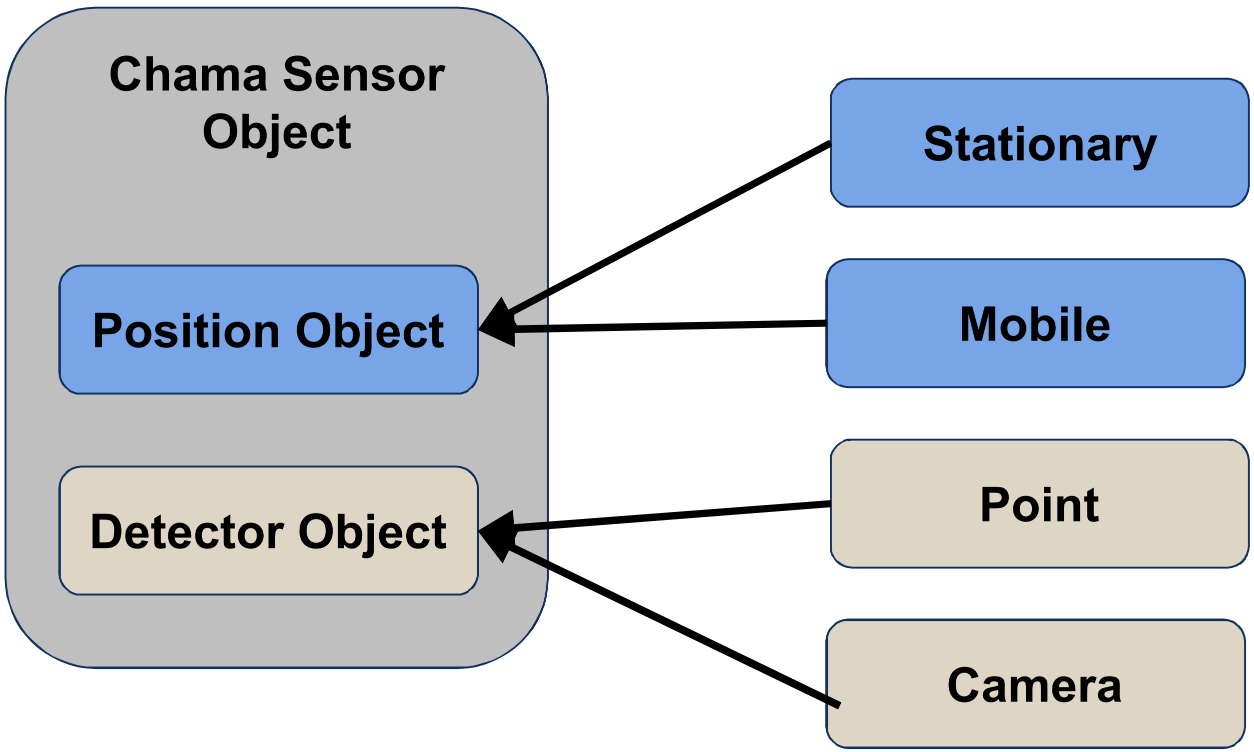
Transport Simulation → **Sensor Technology** → Impact Assessment → Optimization → Graphics

- Stationary and mobile sensors
- Point detectors and cameras
- Detection threshold
- Sensor cost
- Sample times
- Feasible locations or paths
- Failure rates



Sensor Placement Framework

Sandia National Laboratories



CCR Center for Computing Research

Sensor Placement Framework

Transport Simulation → **Sensor Technology** → Impact Assessment → Optimization → Graphics

Mobile Point Sensor

```
>>> pos2 = chama.sensors.Mobile(locations=[(0,0,0),(1,0,0),(1,3,0),(1,2,1)],speed=1.2)
>>> det2 = chama.sensors.Point(threshold=0.001, sample_times=[0,1,2,3,4,5,6,7,8,9,10])
>>> mobile_pt_sensor = chama.sensors.Sensor(position=pos2, detector=det2)
```

Stationary Camera Sensor

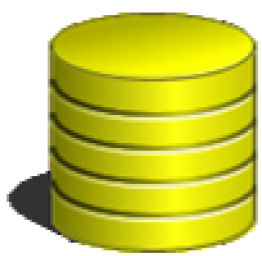
```
>>> pos3 = chama.sensors.Stationary(location=(2,2,1))
>>> det3 = chama.sensors.Camera(threshold=400, sample_times=[0,5,10], direction=(1,1,1))
>>> stationary_camera_sensor = chama.sensors.Sensor(position=pos3, detector=det3)
```

Sensor Placement Framework

Transport Simulation → Sensor Technology → **Impact Assessment** → Optimization → Graphics

- Merging simulation results with sensor technology
  - Thousands of leak scenarios
  - Thousands of potential sensor locations and settings
- Determine how much of the signal is detected by different sensors
- Metrics
  - Time to detection, coverage, etc.

Simulation results
X,Y,Z,T,C



+

Sensor detection

=

Impact assessment

Scenario	Sensor	Impact
A	1	5
A	2	6
B	2	3
...		

Sensor Placement Framework

Transport Simulation → Sensor Technology → **Impact Assessment** → Optimization → Graphics

Define the available sensors

```
>>> sensors = {}
>>> sensors['A'] = stationary_pt_sensor
>>> sensors['B'] = mobile_pt_sensor
>>> sensors['C'] = stationary_camera_sensor
>>> sensors['D'] = mobile_camera_sensor
```

Determine the detection times
(i.e. when a sensor detects each scenario)

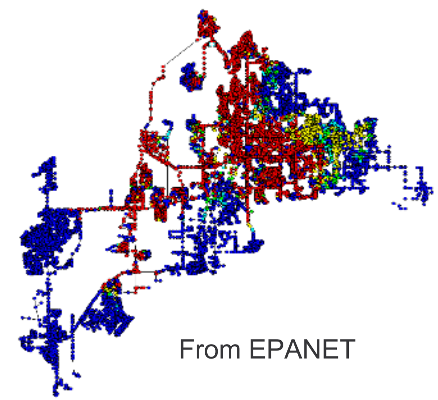
```
>>> det_times = chama.impact.detection_times(signal, sensors)
```

```
>>> print(det_times)
  Scenario Sensor            Impact
0       S1      A              [30]
1       S1      B              [30]
2       S1      C  [10, 20, 30, 40]
3       S2      A      [10, 20, 30]
4       S2      B          [20, 30]
5       S2      C  [10, 20, 30, 40]
6       S3      A          [20, 30]
7       S3      B          [20, 30]
8       S3      C      [20, 30, 40]
```

Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → **Optimization** → Graphics

- Optimization based on 'P-median facilities location'
- Given a sensor budget, determine best combination of sensors to place in the field
- Identify conditions that lead to detected and undetected scenarios
- The methods have proven successful with water security applications



From EPANET

From CALPUFF View

Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → **Optimization** → Graphics

```
>>> print(min_det_time)
  Scenario Sensor  Impact
0       S1      A     2.0
1       S2      A     3.0
2       S3      B     4.0
>>> print(sensor)
  Sensor    Cost
0      A   100.0
1      B   200.0
2      C   500.0
3      D  1500.0
>>> print(scenario)
  Scenario  Undetected Impact  Probability
0       S1               48.0         0.25
1       S2              250.0         0.60
2       S3              100.0         0.15

>>> pmedian = chama.optimize.Pmedian(use_scenario_probability=True, use_sensor_cost=True)
>>> results = pmedian.solve(sensor, scenario, min_det_time, 200)

>>> print(results['Sensors'])
['A']
>>> print(results['Objective']) # 2*0.25+3*0.6+100*0.15
17.3
>>> print(results['Assessment'])
  Scenario Sensor  Impact
0       S1      A     2.0
1       S2      A     3.0
2       S3   None   100.0
```

Formulate and solve P-median formulation

Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → **Optimization** → Graphics

Formulate and solve coverage formulation

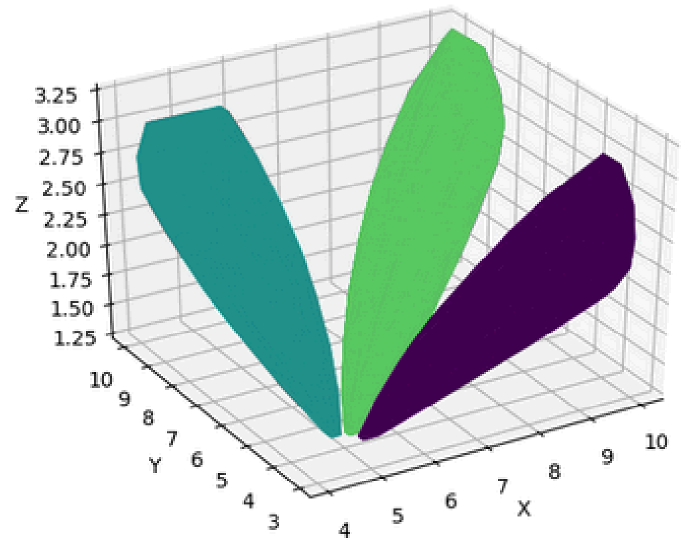
```
>>> print(det_times)
  Scenario Sensor       Impact
0       S1      A    [2, 3, 4]
1       S2      A          [3]
2       S3      B [4, 5, 6, 7]
>>> print(sensor)
  Sensor    Cost
0      A   100.0
1      B   200.0
2      C   500.0
3      D  1500.0
>>> print(scenario)
  Scenario  Undetected Impact  Probability
0       S1               48.0         0.25
1       S2              250.0         0.60
2       S3              100.0         0.15

>>> coverage = chama.optimize.Coverage(use_sensor_cost=True, coverage_type='time')
>>> results = coverage.solve(sensor, scenario, det_times, 200)

>>> print(results['Sensors'])
['B']
>>> print(results['Objective'])
0.5
>>> print(results['Assessment'])
    Scenario Sensor  Impact
0  (4, 'S3')      B     0.0
1  (5, 'S3')      B     0.0
2  (6, 'S3')      B     0.0
3  (7, 'S3')      B     0.0
4  (2, 'S1')   None     1.0
5  (3, 'S1')   None     1.0
6  (3, 'S2')   None     1.0
7  (4, 'S1')   None     1.0
```

Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → Optimization → **Graphics**

Visualize the signal

```
>>> chama.graphics.signal_convexhull(signal, scenarios=['S1', 'S2', 'S3'], threshold=0.01)
```

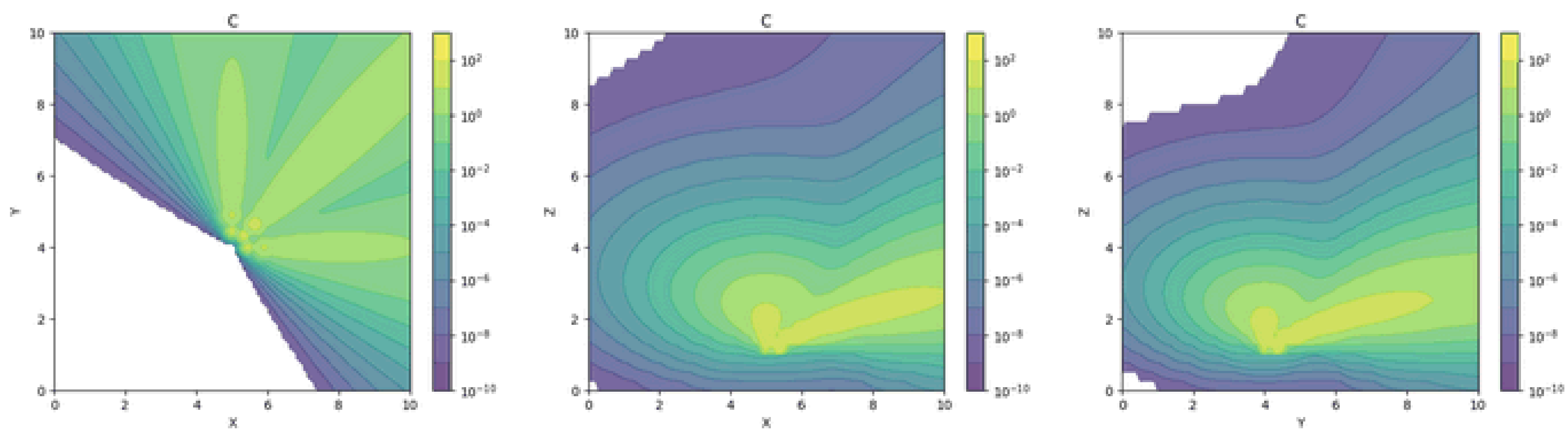


Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → Optimization → **Graphics**

Visualize the signal

```
>>> chama.graphics.signal_xsection(signal, 'S1', threshold=0.01)
```

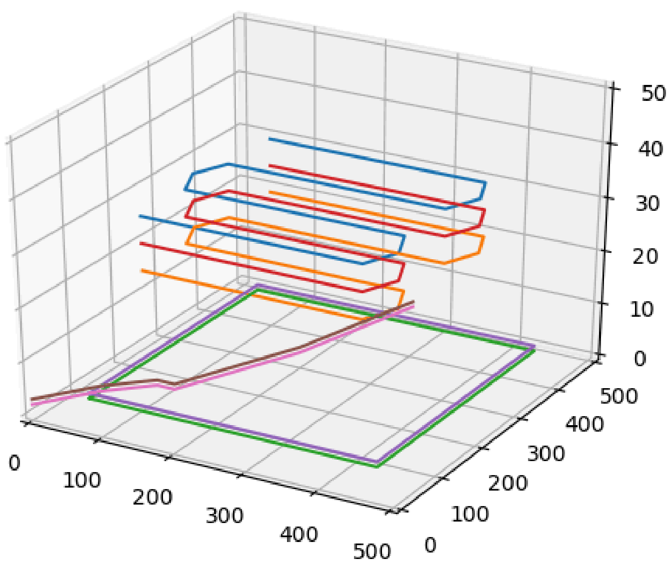


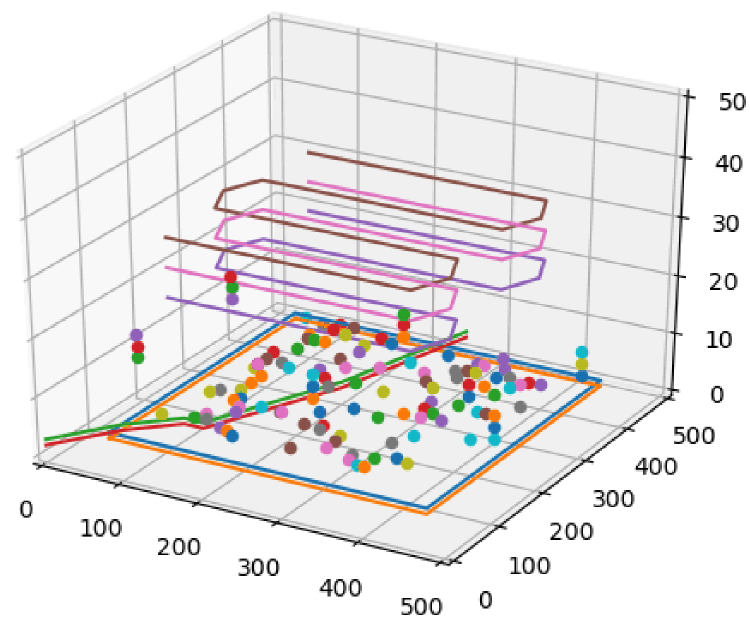
Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → Optimization → **Graphics**

Visualize the sensors

```
chama.graphics.sensors(sensors, x_range=(0,xsize), y_range=(0,ysize), z_range=(0,zsize))
```

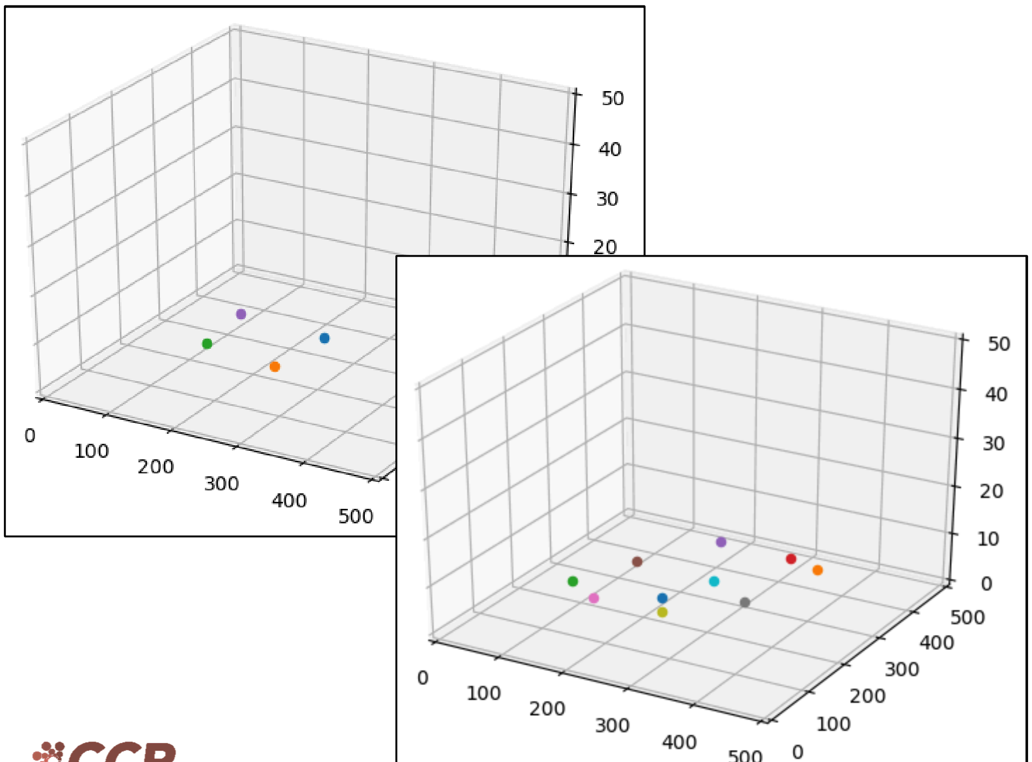


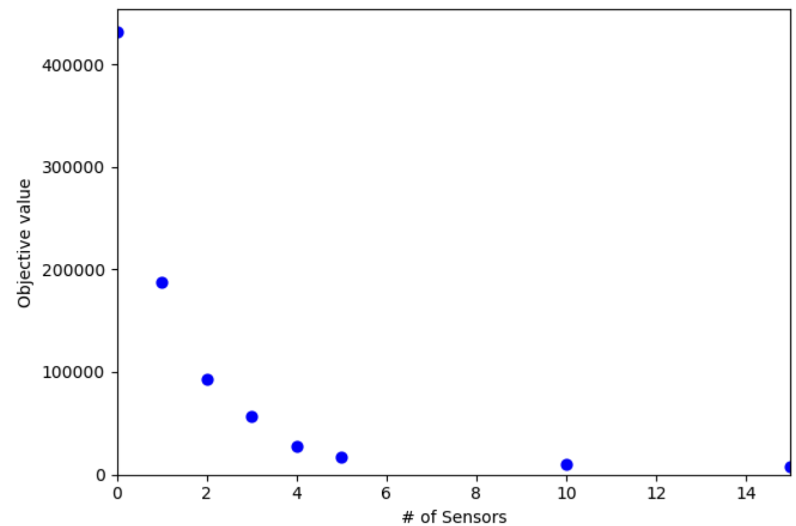


Sensor Placement Framework

Transport Simulation → Sensor Technology → Impact Assessment → Optimization → **Graphics**

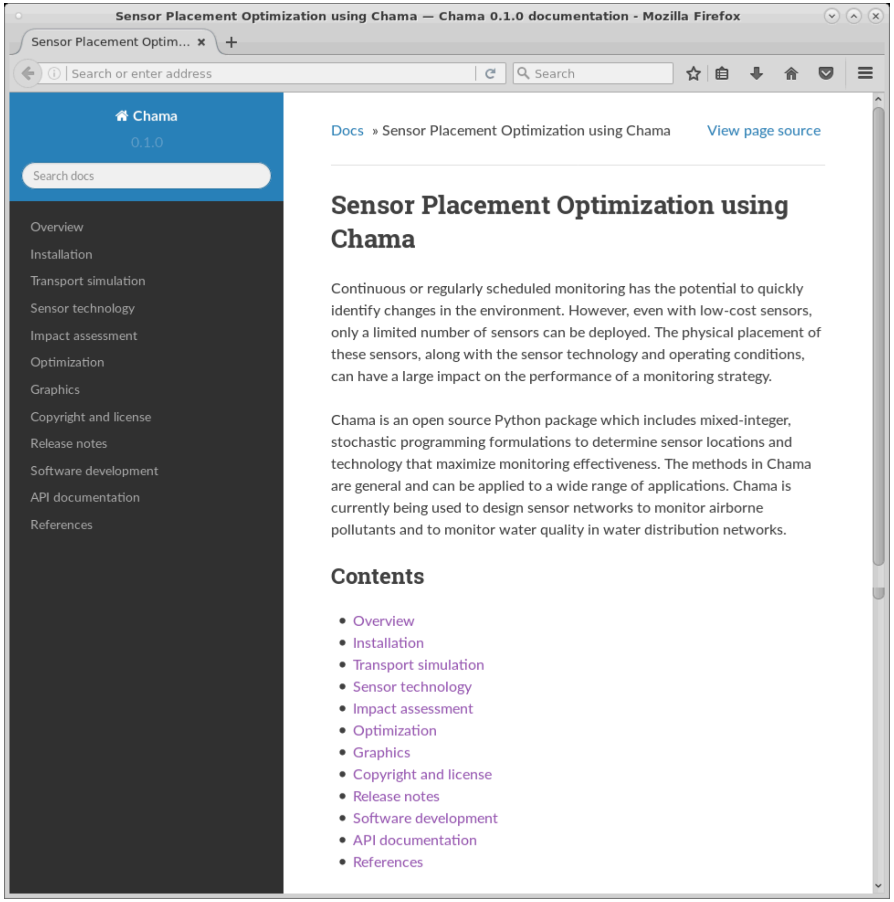
Visualize the results





Summary/Conclusions

- Flexible and extensible framework for sensor placement
- Mix and match modules to support wide variety of applications
- Explore trade-offs between different sensor technologies



Acknowledgements

- Dylan Moriarty, Sandia National Laboratories
- Adam Brandt, Stanford University
- Arvind Ravikumar, Stanford University

This research was supported by the Laboratory Directed Research and Development (LDRD) Program at Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC., a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA-0003525.