Model-Based Prognostics

Indranil Roychoudhury
Prognostics Center of Excellence
Intelligent Systems Division
NASA Ames Research Center, Moffett Field, CA, USA

indranil.roychoudhury@nasa.gov

Definitions

- Prognostics = the problem of predicting the future state of a system
- For our research (and the purposes of this presentation), we are specifically interested in predicting *failure states*
  - EOL = end of life (time of failure)
  - RUL = remaining useful life (time until failure)

Why Prognostics?

- Prognostics can enable:
  - Adopting condition-based maintenance strategies, instead of time-based maintenance
  - Optimally scheduling maintenance
  - Optimally planning for spare components
  - Reconfiguring the system to avoid using the component before it fails
  - Prolonging component life by modifying how the component is used (e.g., load shedding)
  - Optimally plan or replan a mission
- System operations can be optimized in a variety of ways

What is Model-Based Prognostics?

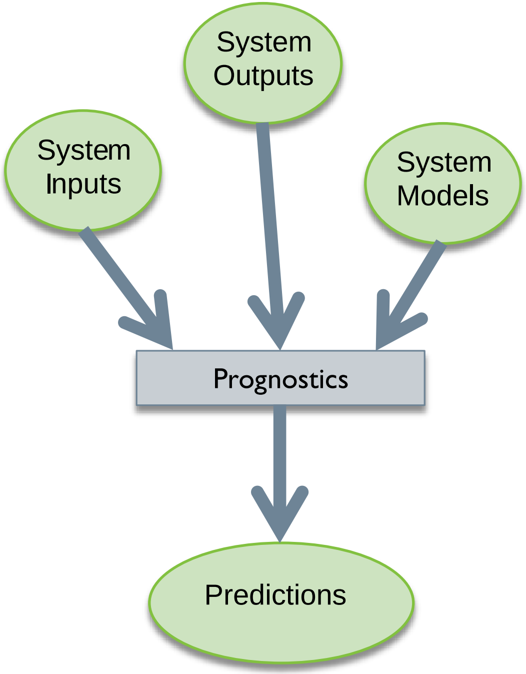
- “Model-based” vs “data-driven”
  - These terms are not very useful! All approaches use models of some kind, and all are driven by data
    - “Model-based” typically refers to approaches using models derived from first principles (e.g., physics-based)
    - “Data-driven” typically refers to approaches using models learned from data (e.g., Artificial Neural Networks, Gaussian Process Regression)
- In practice, models are typically developed from a mix of system knowledge and system data and are typically adapted online in some fashion

Our Definition of Model-Based Prognostics

- Model-based prognostics refers simply to approaches that use mathematical models of system behavior
  - When available, knowledge from first principles, known physical laws, etc, should be used to develop models
  - When a large amount of data is available (for both nominal and degraded behavior), models can be learned from the data
- The general framework will be defined in this context
  - It does not matter how the model was developed
  - It does not matter what the model looks like

Why Model-Based Prognostics?

- With model-based algorithms, models are inputs
  - This means that, given a new problem, we use the same general algorithms
  - Only the models should change
- Model-based prognostics approaches are applicable to a large class of systems, given a model
- Approach can be formulated mathematically, clearly, and precisely



System Inputs

System Outputs

System Models

Prognostics

Predictions

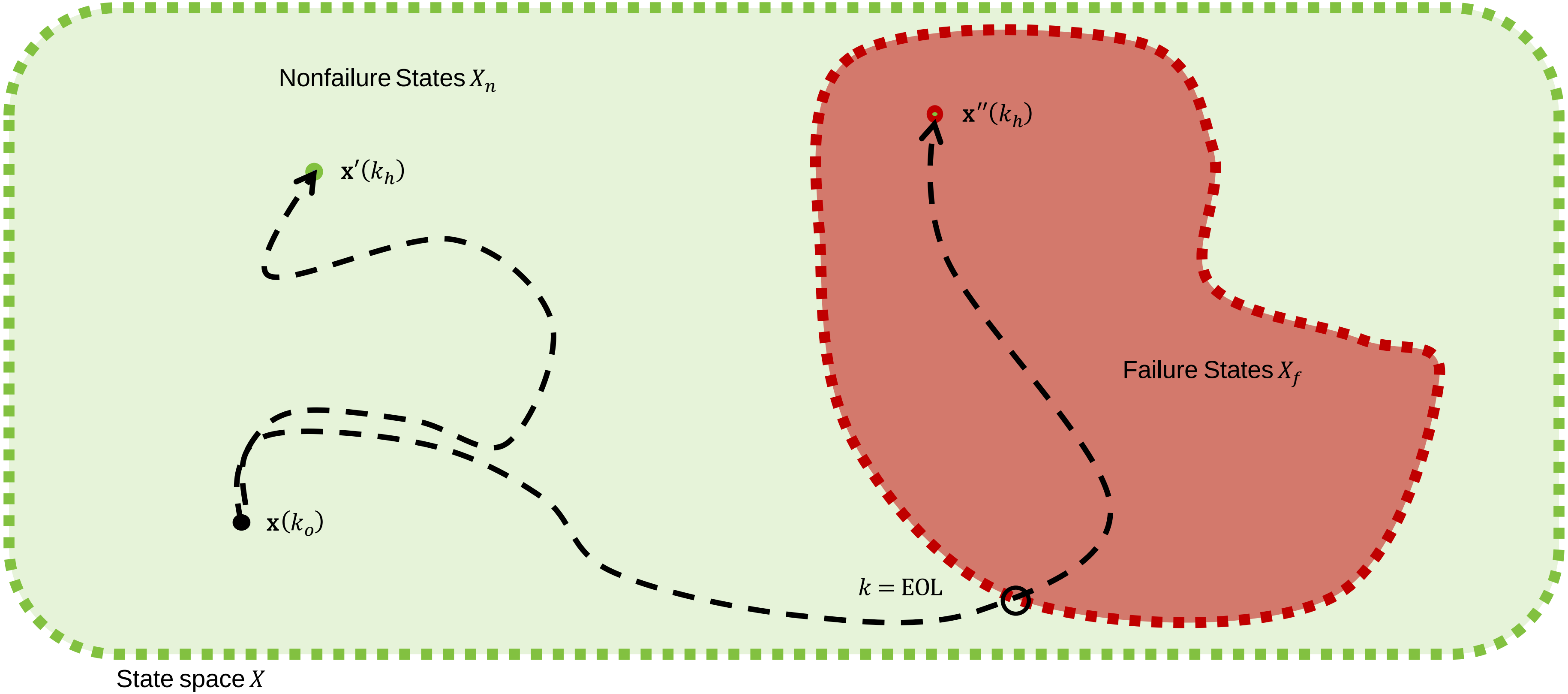
Outline

- **Mathematical Framework**
  - Problem Formulation
  - Computing EOL
  - Handling Uncertainty
  - Probability of Failure
- **Modeling for Prognostics**
  - General Modeling Framework
  - Example:
    - Water Recycling System Prognostics
- **Advanced Prognostics Concepts**
  - System-Level Prognostics
  - Distributed Prognostics
  - Prognostics and Decision Making
  - Examples:
    - Predicting the Safety of our Airspace
    - Rover Mission Replanning
- **Conclusions**

Mathematical Framework

- Problem Formulation
- Computing EOL
- Handling Uncertainty
- Probability of Failure

Concept


Nonfailure States X_n
$\mathbf{x}'(k_h)$
$\mathbf{x}''(k_h)$
Failure States X_f
$\mathbf{x}(k_o)$
$k = \text{EOL}$
State space X

Problem Requirements

- System model
  - System state space
  - Partition into non-failure and failure states
  - System inputs
  - State update equation
- Prediction inputs
  - Initial time k_o
  - Prediction horizon k_h
  - System inputs from k_o to k_h

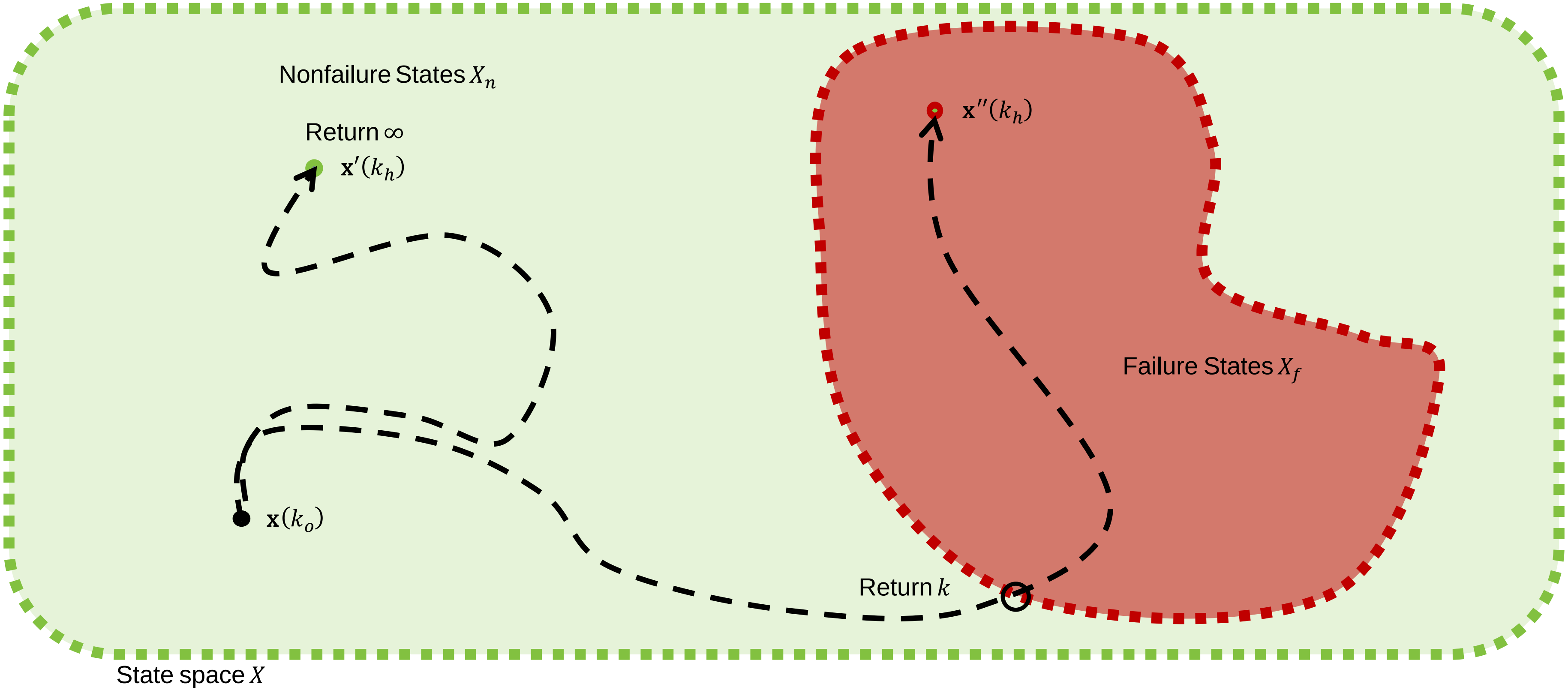
System Model

- Assume system can be modeled using
  - $\mathbf{x}(k+1) = \mathbf{f}\big(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k)\big)$
  - k is the discrete time variable
  - $\mathbf{x}$ is the state vector
  - $\mathbf{u}$ is the input vector
  - $\mathbf{v}$ is the process noise vector
  - $\mathbf{f}$ is the state update equation
- Define a *threshold* function that partitions state-space into nonfailure and failure states
  - $T_f \colon \mathbb{R}^{n_x} \to \{true, false\}$
  - That is, $T_f\big(\mathbf{x}(k)\big)$ returns true when it is a failure state, false otherwise

Initial Problem Formulation

- Assume we know
  - Initial state, $\mathbf{x}(k_o)$
  - Future input trajectory, $\mathbf{U}_{k_o,k_h} = [\mathbf{u}(k_o), \mathbf{u}(k_o + 1), \dots, \mathbf{u}(k_h)]$
  - Process noise trajectory, $\mathbf{V}_{k_o,k_h} = [\mathbf{v}(k_o), \mathbf{v}(k_o + 1), \dots, \mathbf{v}(k_h)]$
- Problem definition
  - Given $k_o, k_h, \mathbf{x}(k_o), \mathbf{U}_{k_o,k_h}, \mathbf{V}_{k_o,k_h}$
  - Compute EOL
    - $\mathrm{EOL}(k) = \inf\left\{k': k' \geq k \text{ and } T_f\left(\mathbf{x}(k)\right)\right\}$

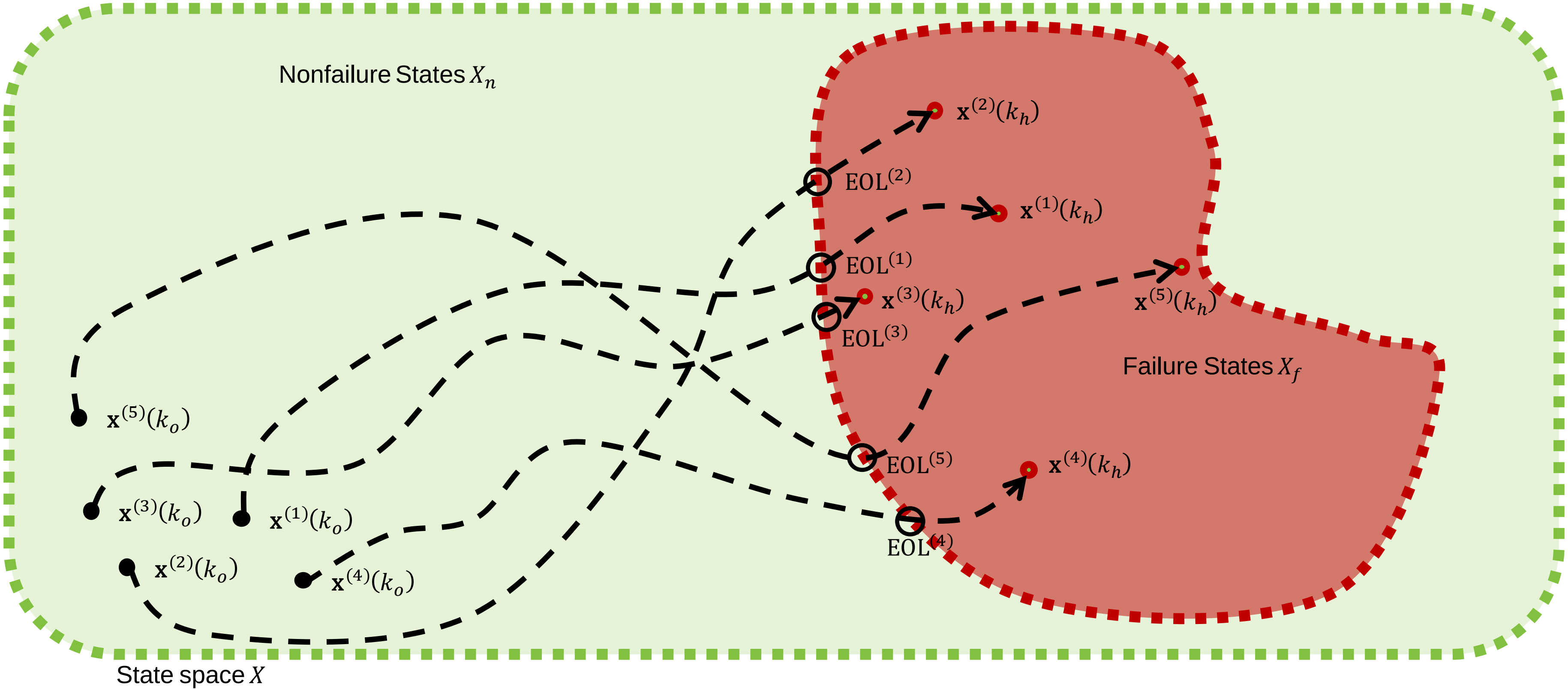
Concept: ComputeEOL



Computational Algorithm

```
ComputeEOL$\left(k_o, k_h, \mathbf{x}(k_o), \mathbf{U}_{k_o,k_h}, \mathbf{V}_{k_o,k_h}\right)$
    $\mathbf{X}_{k_o,k_h}(k_o) \leftarrow \mathbf{x}(k_o)$                                      // Set initial state
    for $k = k_o$ to $k_h - 1$ do
        if $T_f\left(\mathbf{X}_{k_o,k_h}\right)(k)$                                    // Check if failure state
            return $k$                                                  // Return current time as EOL
        end if
        $\mathbf{X}_{k_o,k_h}(k+1) \leftarrow \mathbf{f}\left(\mathbf{X}_{k_o,k_h}(k), \mathbf{U}_{k_o,k_h}(k), \mathbf{V}_{k_o,k_h}(k)\right)$   // Update state
    end for
    if $T_f\left(\mathbf{X}_{k_o,k_h}\right)(k)$                                        // Check if failure state
        return $k$                                                      // Return current time ($k_h$) as EOL
    else
        return $\infty$                                                      // Return infinity
    end if
```

Concept: Uncertainty



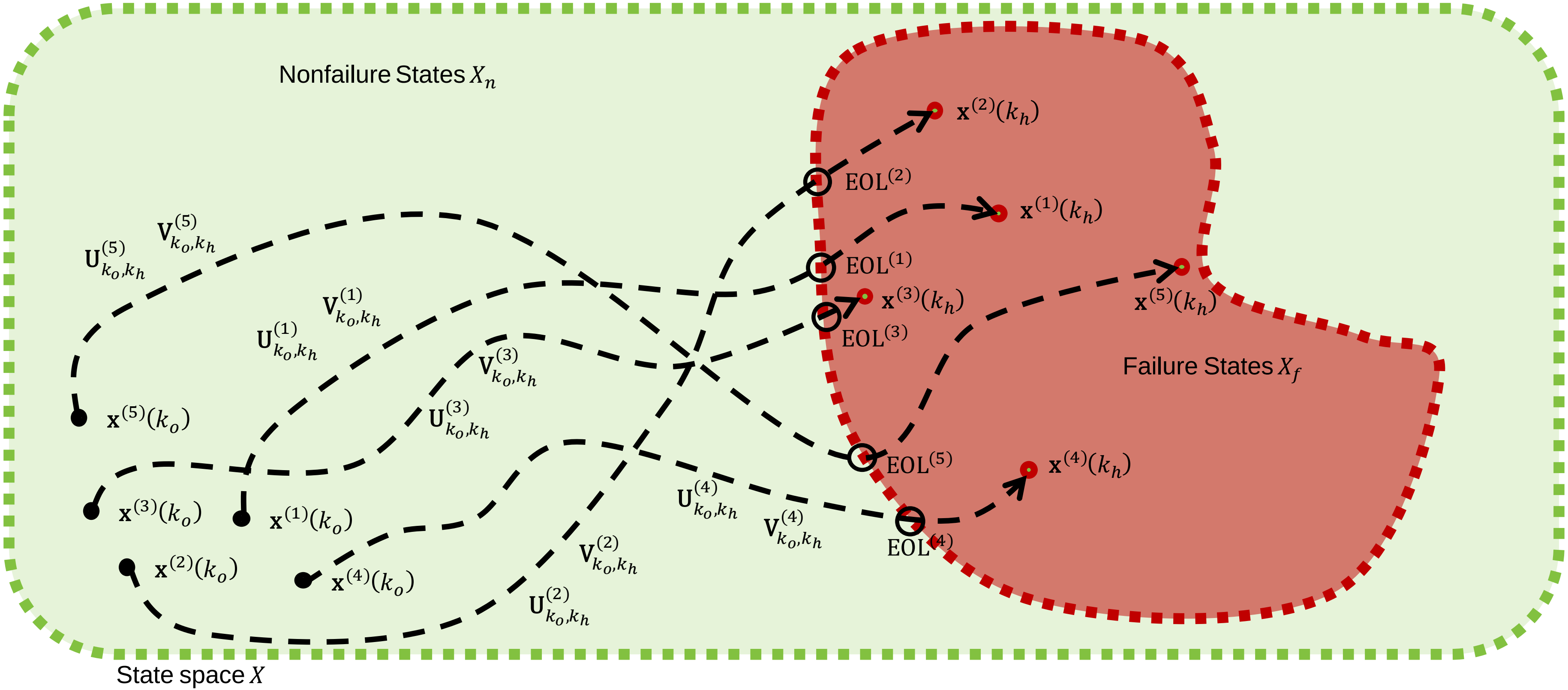
Handling Uncertainty

- Sources of uncertainty
  - Initial state, $\mathbf{x}(k_o)$
  - Future input trajectory, $\mathbf{U}_{k_o,k_h}$
  - Process noise trajectory, $\mathbf{V}_{k_o,k_h}$
- Requirements
  - Must define $p\left(\mathbf{x}(k_o)\right)$
  - Must define $p(\mathbf{U}_{k_o,k_h})$
  - Must define $p(\mathbf{V}_{k_o,k_h})$

Updated Problem Formulation

- Assume we know
  - Initial state distribution, $p\left(\mathbf{x}(k_o)\right)$
  - Future input trajectory distribution, $p(\mathbf{U}_{k_o,k_h})$
  - Process noise trajectory distribution, $p(\mathbf{V}_{k_o,k_h})$
- Problem definition
  - Given $k_o, k_h, p\left(\mathbf{x}(k_o)\right), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h})$
  - Compute $p(\mathrm{EOL}(k_o))$

Concept: ComputePEOL

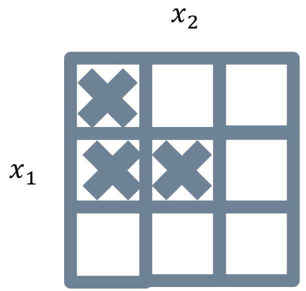


Computational Algorithm

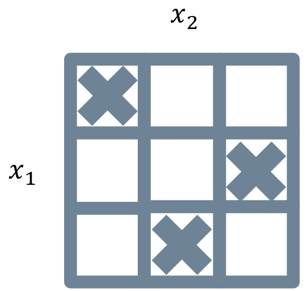
```
$\text{ComputePEOL}(k_o, k_h, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
    // Sample prediction inputs
    $\{(\mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)})\}_{i=1}^{N} \leftarrow \text{GenerateSamples}(N, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
    for $i = 1$ to $N$ do
        // Compute EOL
        $\text{EOL}^{(i)} \leftarrow \text{ComputeEOL}(k_o, k_h, \mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)}$
    end for
    // Return EOL realizations
    return $\{\text{EOL}^{(i)}\}_{i=1}^{N}$
```

Generating Samples

- Compute a set of future state trajectories given realizations of all the uncertain inputs
  - Each state trajectory corresponds to a different EOL prediction
- Sampling algorithms
  - Monte Carlo sampling
    - Independently randomly sample from each of the uncertain distributions N times
    - Typically the required value of N for good results increases nonlinearly with the number of random variables
  - Latin hypercube sampling
    - Near-random sampling in which we ensure each equally probable subspace in each dimension are covered exactly once
    - Guarantees ensemble of samples is representative of real variability
  - Unscented transform sampling
    - Deterministically sample "sigma points" using unscented transform such that statistical moments (e.g., mean and variance) are retained
    - Number of required points scales linearly with the number of random variables
- If samples are weighted, ComputePEOL can be extended accordingly



MC Sampling



LH Sampling

Initial State Distribution

- In general, infer from system knowledge and/or sensor data
- In the Bayesian prognostics paradigm, this is provided by an observer
  - Kalman filter: represents linear system using mean vector and covariance matrix
  - Extended Kalman filter: represents nonlinear system using mean vector and covariance matrix
  - Unscented Kalman filter: represents using weighted sigma point set
  - Particle filter: represents using weighted sample set
- From each of these representations, we can sample realizations for prediction

Process Noise Trajectory Distribution

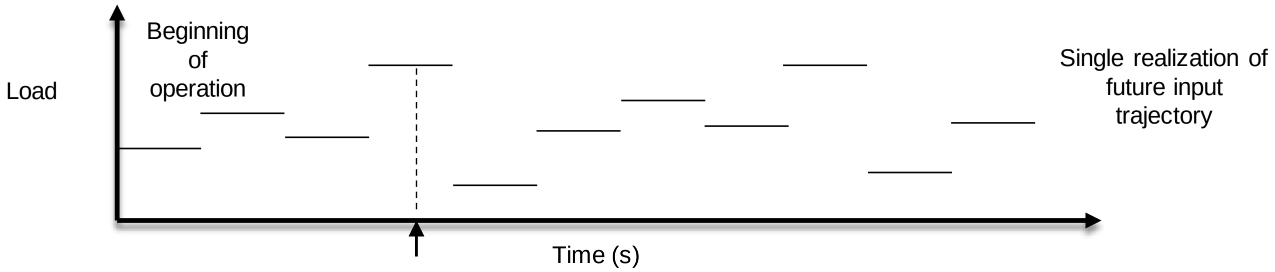
- Assume that process noise at time k is independent of process noise at time k'
  - Represent $\mathbf{v}(k)$ using some assumed distribution
  - For example, multi-variate normal distribution as represented by mean vector and covariance matrix
  - Often assume time-invariant, zero-mean, zero cross-covariance:
    - Example: $\boldsymbol{\mu}_v = [0, 0, \dots, 0], \boldsymbol{P}_{vv} = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix}$
- To construct a realization of $\mathbf{V}_{k_o,k_h}$, independently sample from $p\big(\mathbf{v}(k)\big)$ for each $k \in [k_o, k_h]$

Future Input Trajectory Distribution

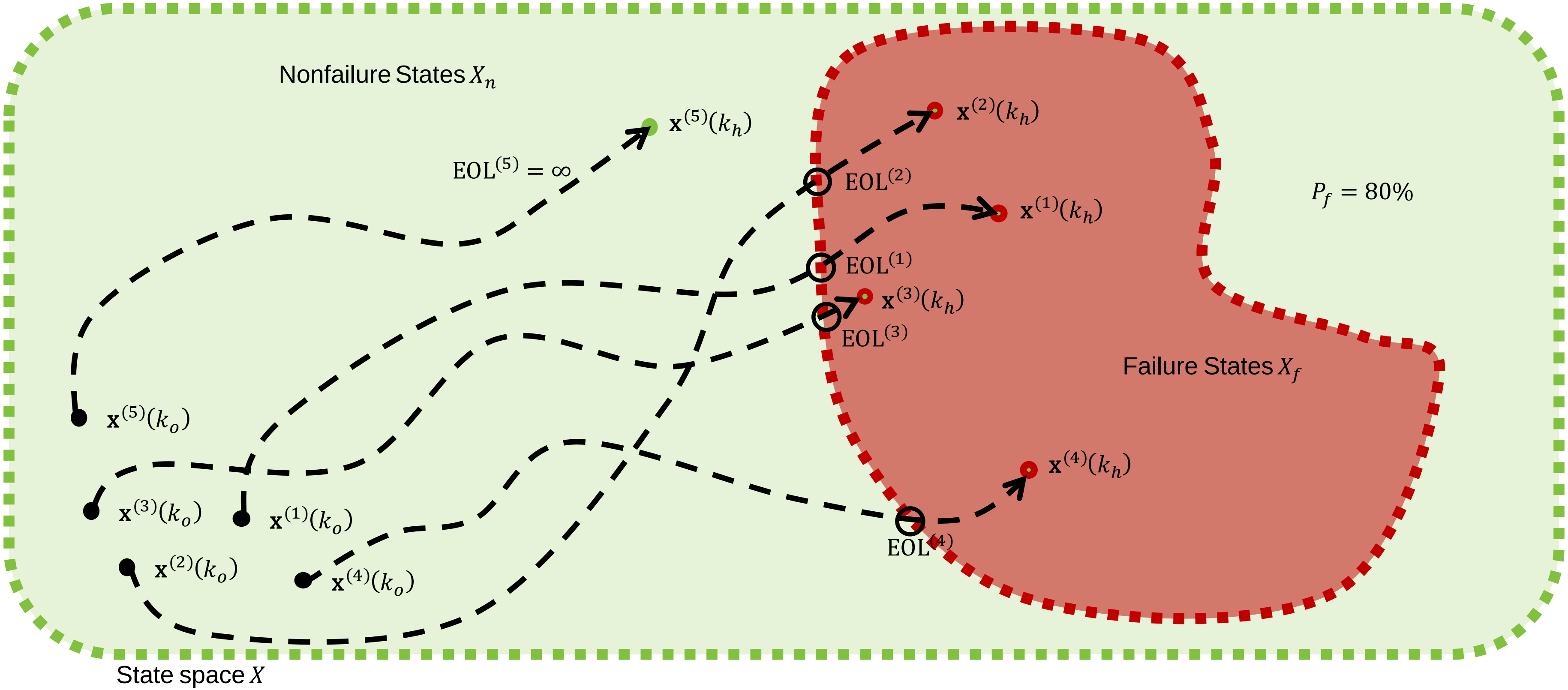
- Input trajectories can take on many different forms, depending on the system
  - Need to obtain values for system inputs for each $k \in [k_o, k_h]$
  - Have potentially $(k_h - k_o + 1) \cdot n_u$ variables to sample!
  - Must reduce dimensionality
- "Surrogate variable" approach
  - Represent a future input trajectory through a small *finite* set of "surrogate variables" that describe how to "construct" an input trajectory
  - Indirectly sample input trajectories by sampling surrogate variables

Future Input Trajectories: Examples

- Constant Load
  - $u(k) = \theta$ for all k, where θ is drawn from a known distribution
- Sinusoidal Load
  - $u(k) = \theta_1 \sin(\theta_2 k)$ for all k, where θ_1 and θ_2 are drawn from known distributions
- Variable Load as Constant Load Segments
  - N total segments
  - Each segment defined by magnitude and duration with known distributions: $2N$ total random variables
  - To sample a trajectory, sample magnitude and duration for each segment



Concept: Probability of Failure

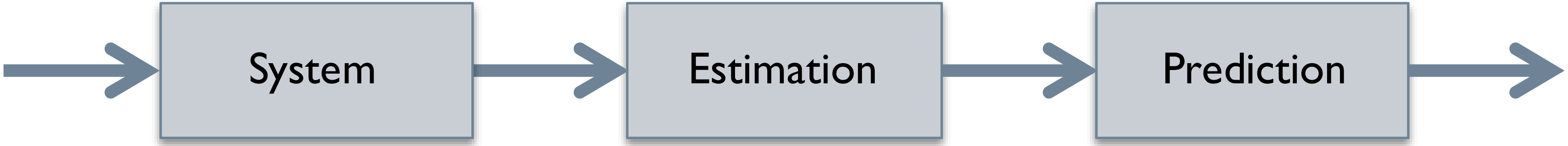


Probability of Failure

- Can compute probability of reaching failure within the given *finite* time horizon within this framework
    - $P_f = P(\mathbf{x}(k_h) \in X_f)$, assuming that: $\mathbf{x}(k) \in X_f \vdash \mathbf{x}(k+1) \in X_f$

```
ComputePf$\left(k_o, k_h, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h})\right)$
    // Sample prediction inputs
    $\{(\mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)})\}_{i=1}^{N} \leftarrow$ GenerateSamples$(N, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
    for $i = 1$ to $N$ do
      // Compute EOL
      $\mathrm{EOL}^{(i)} \leftarrow$ ComputeEOL$(k_o, k_h, \mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)}$
    end for
    // Return EOL realizations
    return $\left|\{\mathrm{EOL}^{(i)} : \mathrm{EOL}^{(i)} < \infty\}\right|_{i=1}^{N} / N$
```

Online Prognostics



- Up to now, described the problem of making a prediction at a single time point
- In online prognostics, predictions are made at several time points
- At each new time step k
  - Update state estimate, $p\left(\mathbf{x}(k)\right)$
    - Use state estimation algorithm (e.g., particle filter)
    - Requires output equation: $\mathbf{y}(k) = \mathbf{h}\left(\mathbf{x}(k), \mathbf{u}(k), \mathbf{n}(k)\right)$, where $\mathbf{n}(k)$ is sensor noise
  - Update future input trajectory distribution, $p(\mathbf{U}_{k,k+h})$
    - May update based on new load schedule
    - May be a function of the state
  - Update process noise trajectory distribution, $p(\mathbf{V}_{k,k+h})$
    - Usually, we assume this is independent of k

Generalizing the Framework

- Partition does not have to be into non-failure/failure states
  - Example: battery not discharged vs. discharged
- Partition does not have to be among only two types
  - Can make predictions w/r/t different state "labels"
- Does not have to be a partition
  - Some states can be assigned multiple "labels"
- This is a framework for predicting time of event occurrence and probability of events

Damage Modeling Framework

- Damage variables, $\mathbf{d}(k) \subseteq \mathbf{x}(k)$
- Damage progression equations
  - $\dot{d}_1 = f_1\big(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k)\big)$
  - $\dot{d}_2 = f_2\big(\mathbf{x}(k), \mathbf{u}(k), \mathbf{v}(k)\big)$
  - $\ldots$
- Damage progression parameters
  - E.g., $\dot{d}_1 = w_1 \cdot x_1(k) + w_2 \cdot x_2(k) \cdot u_1 \ (k)$
  - w_1 and w_2 parameterize the damage progression equation
  - Typically, only the order of magnitude is known, and they must be estimated online along with the states

Damage Modeling Procedure

- Develop nominal model
- Identify model parameters that may change as a function of damage
  - These become the damage variables
  - They augment the state vector of the nominal model
- Develop damage progression equations describing how these variables evolve in time
  - Determine damage progression parameters
  - Determine range of values for expected EOL

A Case Study

- Water Recycling System Prognostics

Example: The Water Recycling System

- The ability to recycle potable water from waste water is an integral part of the Environmental Control and Life Support System (ECLSS) of human-rated space missions
- The forward-osmosis water recycling system (WRS) installed in the "Sustainability Base"
  - The "Sustainability Base" is a new Leadership in Energy and Environmental Design (LEED) Platinum certified "green" office building at ARC
  - The WRS recycles all of the water from the sinks and showers in the building to potable water
  - Designed to reduce water consumption by 60%
- The WRS permits long-duration testing of next-generation WRS technology that is under consideration for use in Deep Space Habitats

Sustainability Base

The Water Recycling System

Why Prognostics for WRS?

- The WRS is a complex engineered system with a large number of components
  - Subject to degradation due to regular use
  - Faults
- Prognosis applications can be implemented to enable condition-based maintenance
  - Avoid unplanned outages
  - Extend the useful life of the system
- Having a good estimate of when things fail is useful for planning for maintenance tasks
  - Performing maintenance too early results in low utilization of resources
  - Performing maintenance action after failure results in system downtime

Forward and Reverse Osmosis

- The WRS consists of a Forward Osmosis (FO) module + a Reverse Osmosis (RO) module
- FO is the movement of solvent molecules across a semi-permeable membrane from a region of higher water chemical potential to a region of lower water chemical potential
  - Driven by the difference in solute concentrations across the membrane
- RO is the movement of solvent molecules across a semi-permeable membrane in the opposite direction of FO, i.e., from a region of lower water chemical potential to a region of higher water chemical potential due to the application of hydraulic pressure
  - Driven by the application of external pressure
- Forward osmosis: $\Delta P = 0$
- Reverse osmosis: $\Delta P > \pi$

Water flux

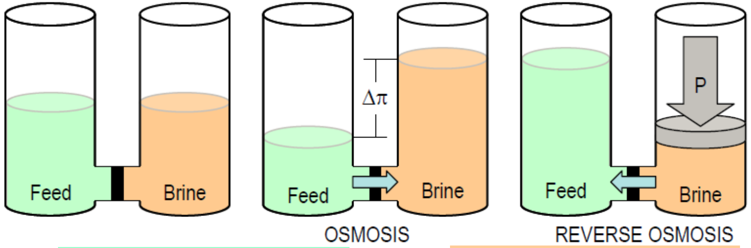
Osmotic pressure differential

Applied pressure differential

$$J_w = A(\sigma\Delta\pi - \Delta P)$$

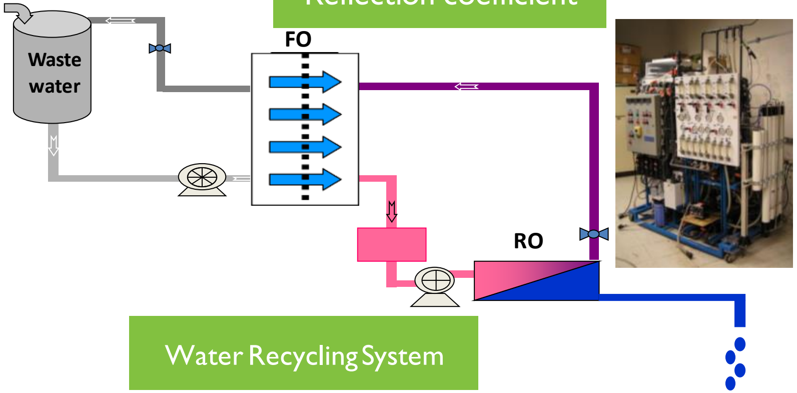
Water permeability constant

Reflection coefficient



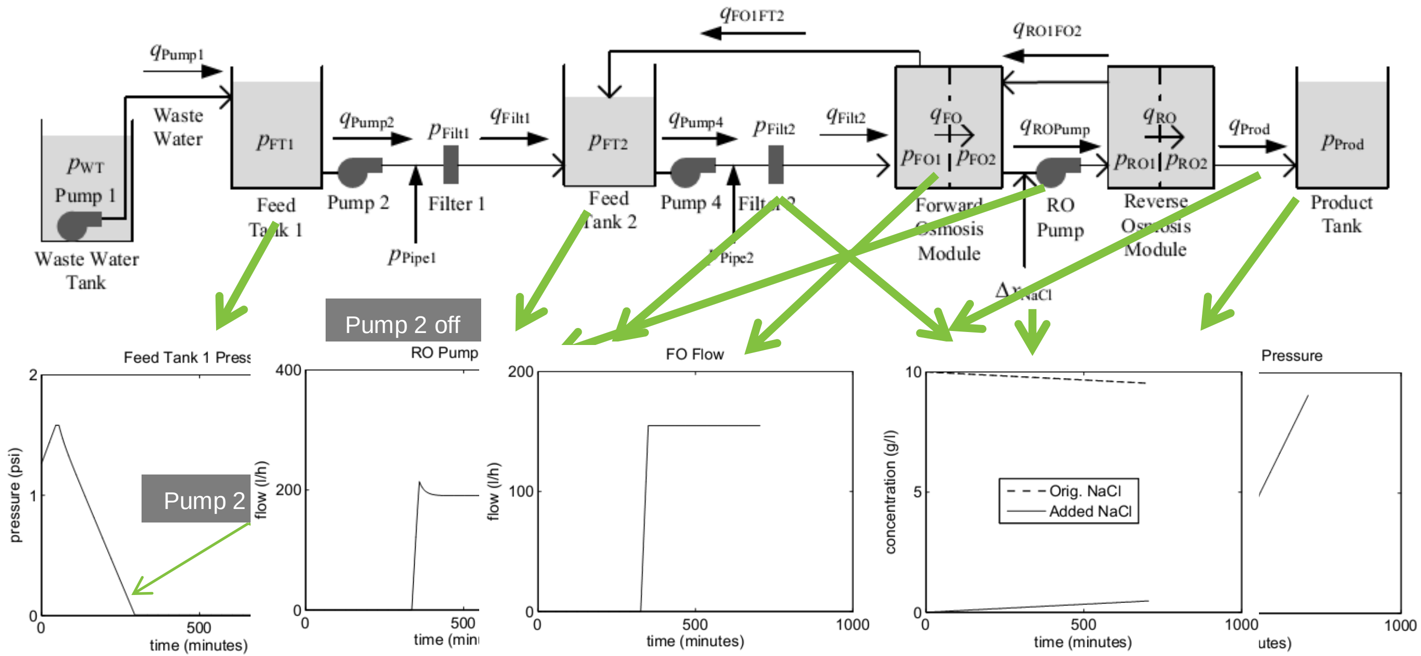
Lower concentration

Higher concentration



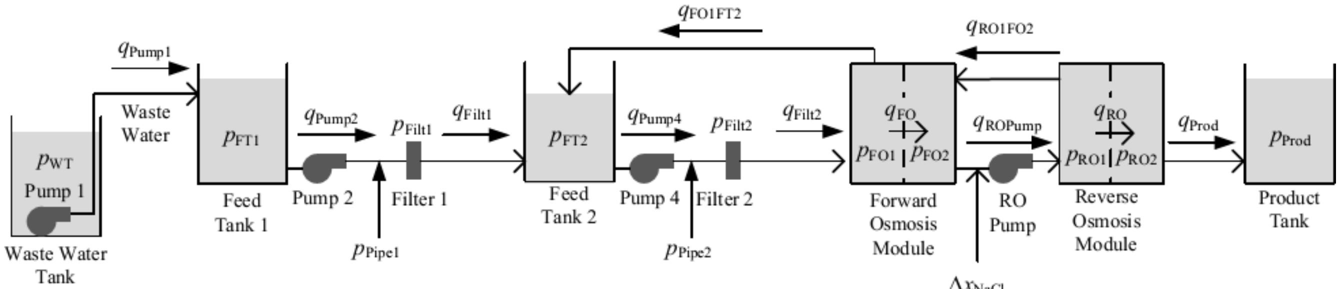
Water Recycling System

Nominal WRS Dynamics



$$f_{FO} = Area_{Memb_{FO}} J_w = Area_{Memb_{FO}} A(\sigma\Delta\pi - \Delta P)$$

Nominal WRS Equations



$$\dot{p}_{\mathrm{WT}} = \frac{1}{C_{\mathrm{WT}}}(-q_{\mathrm{Pump1}})$$

$$\dot{p}_{\mathrm{FT1}} = \frac{1}{C_{\mathrm{FT1}}}(q_{\mathrm{Pump1}} - q_{\mathrm{Pump2}})$$

$$\dot{p}_{\mathrm{Pipe1}} = \frac{1}{C_{\mathrm{Filt1}}}(q_{\mathrm{Pump2}} - q_{\mathrm{Filt1}})$$

$$\dot{p}_{\mathrm{FT2}} = \frac{1}{C_{\mathrm{FT2}}}(q_{\mathrm{Filt1}} + q_{\mathrm{FO1FT2}} - q_{\mathrm{Pump4}})$$

$$\dot{p}_{\mathrm{Pipe2}} = \frac{1}{C_{\mathrm{Filt2}}}(q_{\mathrm{Pump4}} - q_{\mathrm{Filt2}})$$

$$\dot{p}_{\mathrm{FO1}} = \frac{1}{C_{\mathrm{FO1}}}(q_{\mathrm{Filt2}} - q_{\mathrm{FO1FT2}} - q_{\mathrm{FO}})$$

$$\dot{p}_{\mathrm{FO2}} = \frac{1}{C_{\mathrm{FO2}}}(q_{\mathrm{FO}} + q_{\mathrm{RO1FO2}} - q_{\mathrm{ROPump}})$$

$$\dot{p}_{\mathrm{RO1}} = \frac{1}{C_{\mathrm{RO1}}}(q_{\mathrm{ROPump}} - q_{\mathrm{RO1FO2}} - q_{\mathrm{RO}})$$

$$\dot{p}_{\mathrm{RO2}} = \frac{1}{C_{\mathrm{RO2}}}(q_{\mathrm{RO}} - q_{\mathrm{Prod}})$$

$$\dot{p}_{\mathrm{Prod}} = \frac{1}{C_{\mathrm{Prod}}}(q_{\mathrm{Prod}})$$

$$\Delta_{x_{\mathrm{NaCl}}} = min(20, \frac{155 \times 2.78 \times 10^{-8}}{0.841 \times 10^5 \times A_{\mathrm{FO}}} - x_{\mathrm{NaCl}})$$

$$q_{\mathrm{Pump1}} = u_{\mathrm{Pump1}}(R_{\mathrm{Pump1}}\sqrt{|p_{\mathrm{WT}} + p_{\mathrm{Pump1}} - p_{\mathrm{FT1}}|}\mathrm{sign}(p_{\mathrm{WT}} + p_{\mathrm{Pump1}} - p_{\mathrm{FT1}}))$$

$$q_{\mathrm{Pump2}} = u_{\mathrm{Pump2}}(R_{\mathrm{Pump2}}\sqrt{|p_{\mathrm{FT1}} + p_{\mathrm{Pump2}} - p_{\mathrm{Filt1}}|}\mathrm{sign}(p_{\mathrm{FT1}} + p_{\mathrm{Pump2}} - p_{\mathrm{Filt1}}))$$

$$q_{\mathrm{Filt1}} = R_{\mathrm{Filt1}}\sqrt{|p_{\mathrm{Pipe1}} - p_{\mathrm{FT2}}|}\mathrm{sign}(p_{\mathrm{Pipe1}} - p_{\mathrm{FT2}})$$

$$q_{\mathrm{Pump4}} = u_{\mathrm{Pump4}}(R_{\mathrm{Pump4}}\sqrt{|p_{\mathrm{FT2}} + p_{\mathrm{Pump4}} - p_{\mathrm{Filt2}}|}\mathrm{sign}(p_{\mathrm{FT2}} + p_{\mathrm{Pump4}} - p_{\mathrm{Filt2}}))$$

$$q_{\mathrm{FO1FT2}} = R_{\mathrm{FO1FT2}}\sqrt{|p_{\mathrm{FO1}}|}\mathrm{sign}(p_{\mathrm{FO1}})$$

$$q_{\mathrm{Filt2}} = R_{\mathrm{Filt2}}\sqrt{|p_{\mathrm{Pipe2}} - p_{\mathrm{FO1}}|}\mathrm{sign}(p_{\mathrm{Pipe2}} - p_{\mathrm{FO1}})$$

$$q_{\mathrm{FO}} = u_{\mathrm{FO}} \cdot Area_{\mathrm{FO}} \cdot A_{\mathrm{FO}}(x_{\mathrm{NaCl}} + \Delta_{x_{\mathrm{NaCl}}}) \times 0.841 \times 10^5$$

$$q_{\mathrm{RO1FO2}} = R_{\mathrm{RO1FO2}}\sqrt{|p_{\mathrm{RO1}} - p_{\mathrm{FO2}}|}\mathrm{sign}(p_{\mathrm{RO1}} - p_{\mathrm{FO2}}))$$

$$q_{\mathrm{ROPump}} = u_{\mathrm{ROPump}}(R_{\mathrm{ROPump}}\sqrt{|p_{\mathrm{FO2}} + p_{\mathrm{ROPump}} - p_{\mathrm{RO1}}|}\mathrm{sign}(p_{\mathrm{FO2}} + p_{\mathrm{ROPump}} - p_{\mathrm{RO1}}))$$

$$q_{\mathrm{RO}} = u_{\mathrm{RO}} \cdot Area_{\mathrm{RO}} \cdot A_{\mathrm{RO}}(20 \times 0.841 \times 10^5 - (x_{\mathrm{NaCl}} + \Delta_{x_{\mathrm{NaCl}}}) \times 0.841 \times 10^5)$$

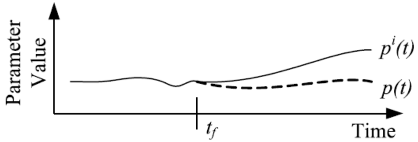
$$q_{\mathrm{Prod}} = R_{\mathrm{Prod}}\sqrt{|p_{\mathrm{RO2}} - p_{\mathrm{Prod}}|}\mathrm{sign}(p_{\mathrm{RO2}} - p_{\mathrm{Prod}})$$

$$p_{\mathrm{Filt1}} = p_{\mathrm{Pipe1}} - p_{\mathrm{FT2}}$$

$$p_{\mathrm{Filt2}} = p_{\mathrm{Pipe2}} - p_{\mathrm{FO1}}$$

Degradation Modeling

- Unexpected change in system components, e.g., filters, membranes, pumps, sensors
  - Modeled as parameter changes
- Faults assumed to be incipient
- Single fault assumption
- Faults are persistent



Incipient Fault

$$\dot{R}_{\text{Filti}} = \begin{cases} 0, & t < t_f \\ \Delta R_{\text{Filt1}}, & \text{otherwise} \end{cases}$$

Filter Blockage

$$q_{\text{Filt2}} = R_{\text{Filt2}} \sqrt{|p_{\text{Pipe2}} - p_{\text{FO1}}|} \text{sign}(p_{\text{Pipe2}} - p_{\text{FO1}}) \qquad (23)$$

$$q_{\text{FO}} = u_{\text{FO}} \cdot Area_{\text{FO}} \cdot A_{\text{FO}}(x_{\text{NaCl}} + \Delta_{x_{\text{NaCl}}}) \times 0.841 \times 10^5 \qquad (24)$$

$$\dot{S} = \begin{cases} 0, & t < t_f \\ \Delta S, & \text{otherwise.} \end{cases}$$

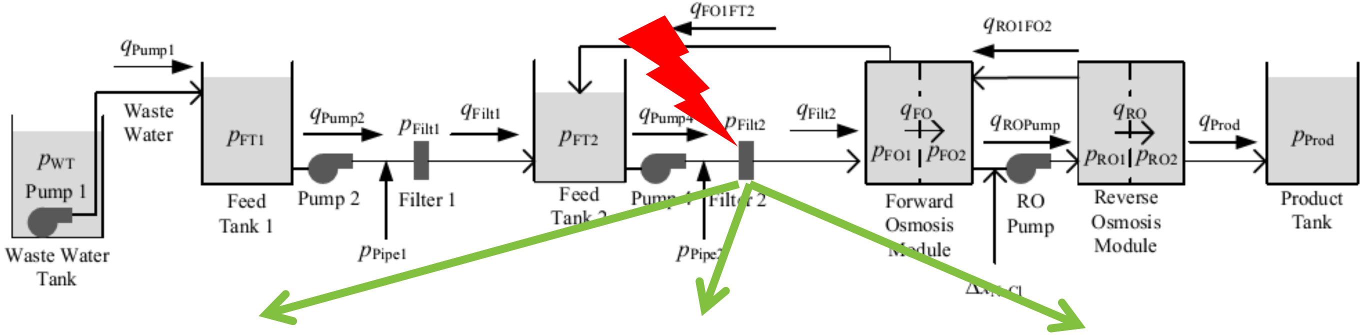
Sensor Drift

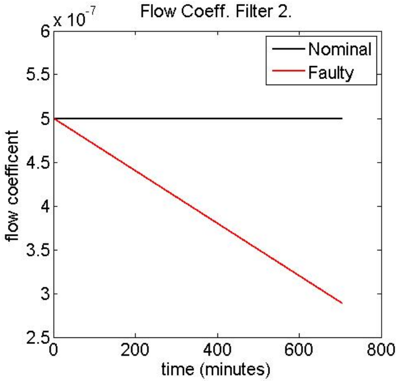
$$\dot{A}_i = \begin{cases} 0, & t < t_f \\ \Delta A_i, & \text{otherwise} \end{cases}$$

Membrane Blockage

Faulty WRS Dynamics: Filter 2 Clogging

$$q_{\text{Filt2}} = R_{\text{Filt2}}\sqrt{|p_{\text{Pipe2}} - p_{\text{FO1}}|}\,\text{sign}(p_{\text{Pipe2}} - p_{\text{FO1}})$$





Prognostics for WRS

```
$\text{ComputePEOL}(k_o, k_h, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
    // Sample prediction inputs
    $\{(\mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)})\}_{i=1}^{N} \leftarrow \text{GenerateSamples}(N, p(\mathbf{x}(k_o)), p(\mathbf{U}_{k_o,k_h}), p(\mathbf{V}_{k_o,k_h}))$
    for $i = 1$ to $N$ do
        // Compute EOL
        $\text{EOL}^{(i)} \leftarrow \text{ComputeEOL}(k_o, k_h, \mathbf{x}(k_o)^{(i)}, \mathbf{U}_{k_o,k_h}{}^{(i)}, \mathbf{V}_{k_o,k_h}{}^{(i)}$
    end for
    // Return EOL realizations
    return $\{\text{EOL}^{(i)}\}_{i=1}^{N}$
```

✓

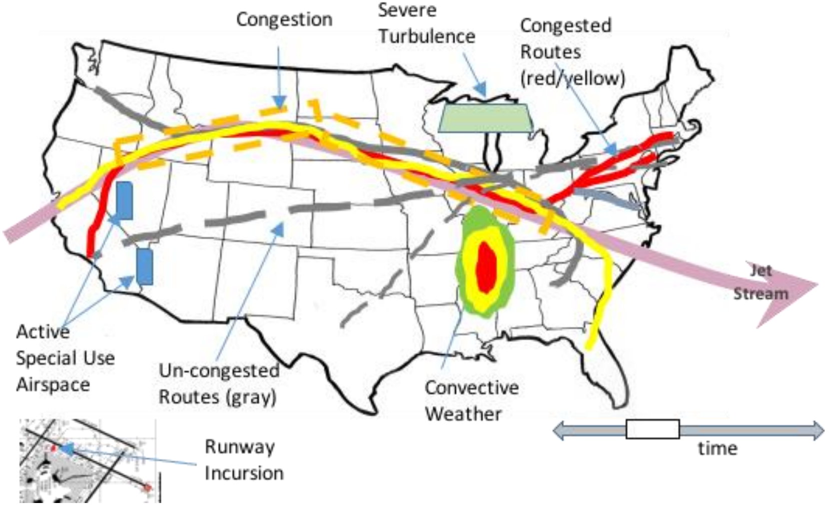
k_o	✓
k_h	✓
Models	✓
$p(\mathbf{x}(k_o))$	✓(Normal Gaussian)
$p(\mathbf{U}_{k_o,k_h})$	✓(Hypothesize Inputs)
$p(\mathbf{V}_{k_o,k_h})$	✓(Normal Gaussian)

Advanced Prognostics Concepts

- System-Level Prognostics
- Distributed Prognostics
- Prognostics and Decision Making
- Examples: Predicting the Safety of our Airspace, Rover Decision Making

Example: Real-Time Safety Monitoring of National Airspace System

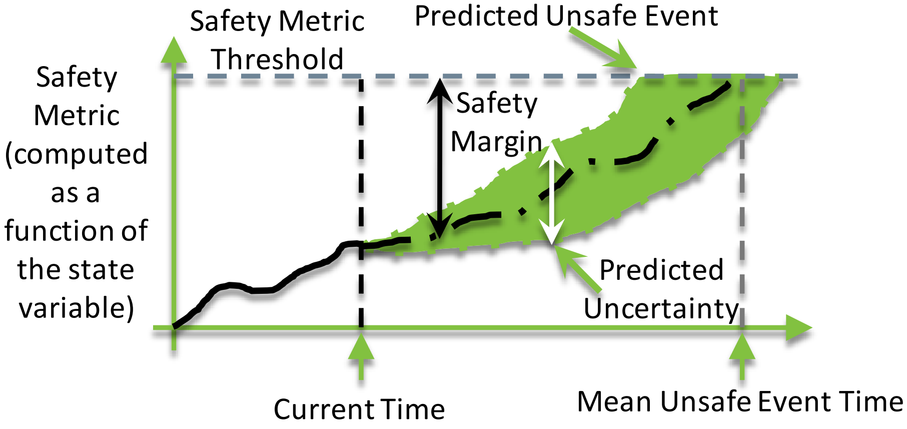
- RTSM framework
  - Provides real-time assessment (nowcast and forecast) of safety and risk to NAS
  - Predicts evolution of safety to proactively avoid unsafe situations instead of reactively mitigating them
  - Holistic framework
    - Generalization of prognostics concepts
    - Combines multiple threats to safety and considers their potential interactions
    - Integrates disparate data sources
    - Incorporates multiple sources of uncertainty into the predictions



Congestion
Severe Turbulence
Congested Routes (red/yellow)
Jet Stream
Active Special Use Airspace
Un-congested Routes (gray)
Convective Weather
Runway Incursion
time

Terminology

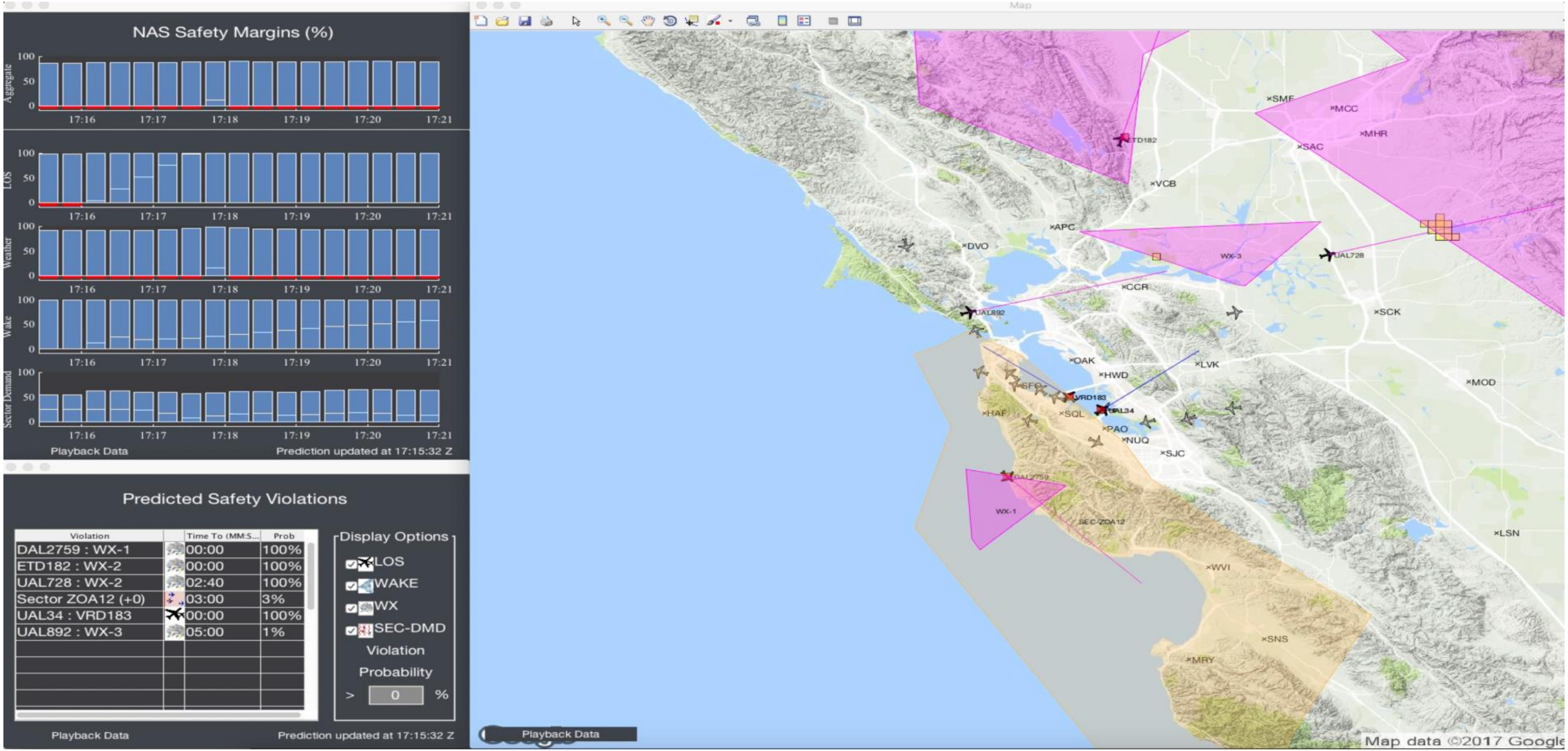
- **Unsafe event**:An event/situation that compromises NAS safety or established safety standards
  - Examples: loss of separation, loss of control, controlled flight into terrain, runway incursion, hard landing, tail strike, collision, etc.
- **Safety metric**:A quantitative measure of some aspect of safety of the NAS
  - Examples: distance between two aircraft, distance between aircraft and convective weather region
- **Safety threshold**: Some limit on a safety metric or set of safety metrics
  - Example: Enroute separation of 5 nautical miles
- **Safety margin**:"Distance" between current safety metric(s) and safety threshold(s)



Approach Overview

- **Model-based approach -** require dynamic models
  - Model aircraft, weather regions, etc.
  - Predictions improve with more accurate models
  - Tradeoff between model fidelity and computational performance
- **Capture uncertainty** inherent in
  - Sensor information (sensor noise, message delay, etc.), system models, and system inputs (e.g., aircraft intent information)
- **Identify categories of events** to model
  - Loss of separation, wake vortex encounter, convective weather encounter, sector demand violation, etc.
- Determine what conditions define the occurrence of each event
  - Defined using **safety metrics which are functions of the NAS state**
  - Example: Loss of separation between A1 and A2 occurs when the horizontal separation is less than 5 nautical miles and the vertical separation is less than 1000 ft
  - Example: Sector demand is too high when the number of aircraft in a sector meets or exceeds the capacity limit
- **Compute the safety margin** w/r/t an event
  - Margin computed as "distance" to event threshold, over threshold, in [0,100]%
  - Margin is 0% when event is present
- **Compute aggregate safety margins**
  - Average safety margins over all potential events

RTSM – System-level Prognostics Example



Predicted Safety Violations

Violation		Time To (MM:S...	Prob
DAL2759 : WX-1		00:00	100%
ETD182 : WX-2		00:00	100%
UAL728 : WX-2		02:40	100%
Sector ZOA12 (+0)		03:00	3%
UAL34 : VRD183		00:00	100%
UAL892 : WX-3		05:00	1%

Display Options

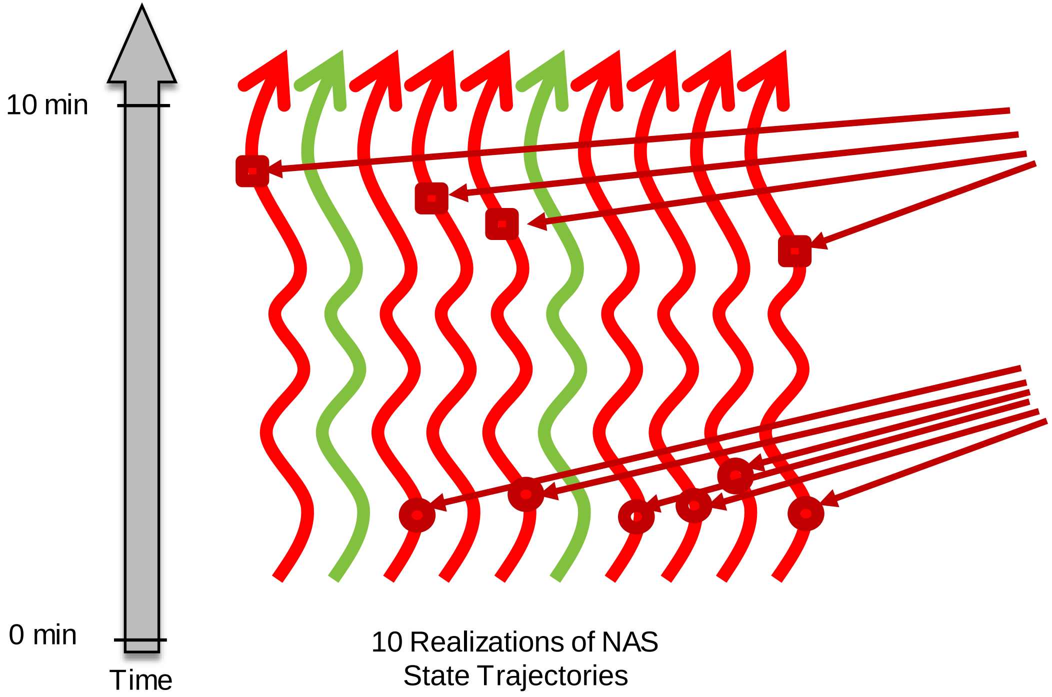
☑ LOS
☑ WAKE
☑ WX
☑ SEC-DMD

Violation Probability > 0 %

Playback Data

Prediction updated at 17:15:32 Z

RTSM – System-level Prognostics Example



Occurrences of $WX_{W1,A3}$:

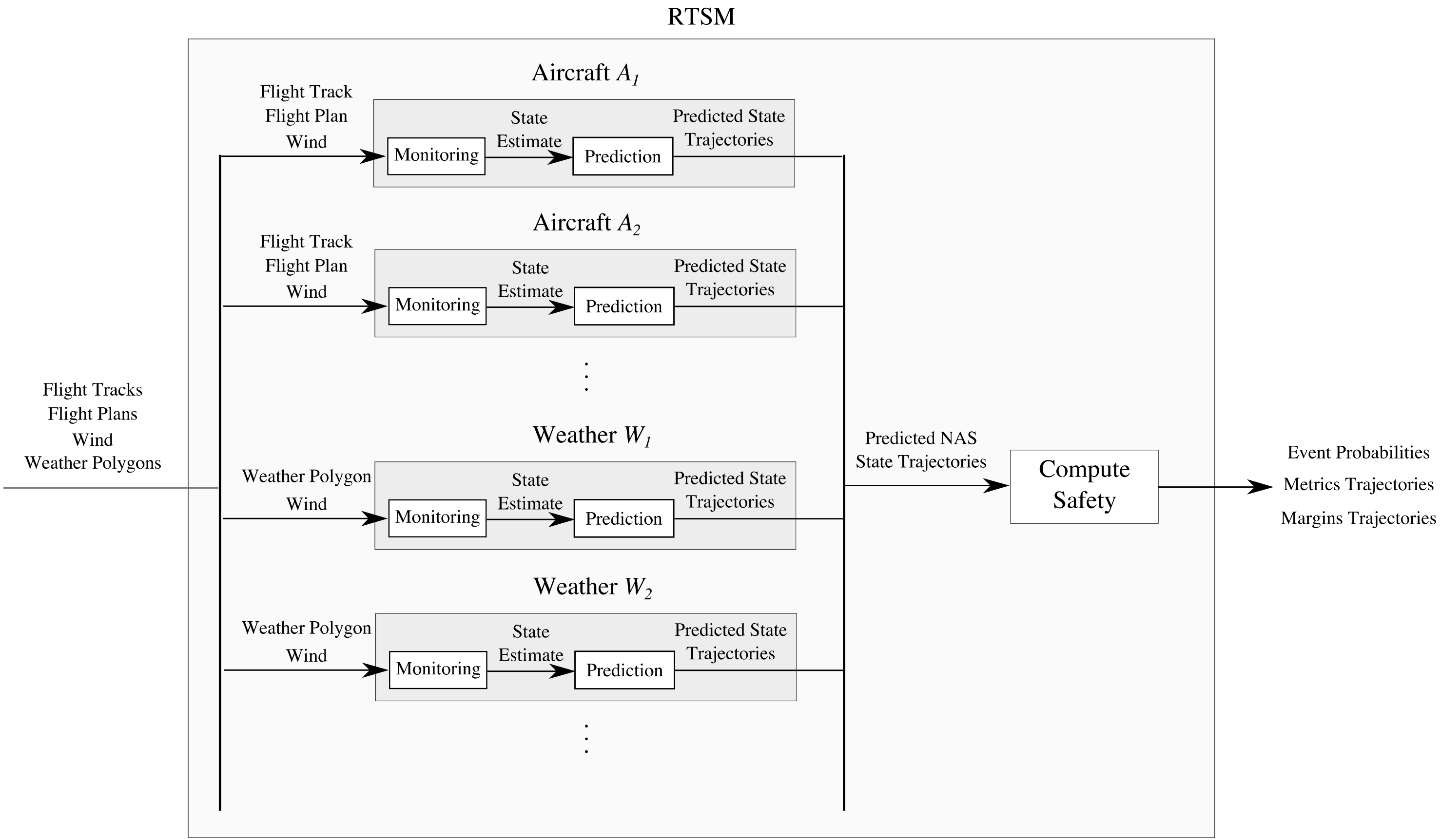
1. Probability = 40%
2. Time until event = 8 min. (average)

Occurrences of $LOS_{A1,A2}$:

1. Probability = 60%
2. Time until event = 2 min. (average)

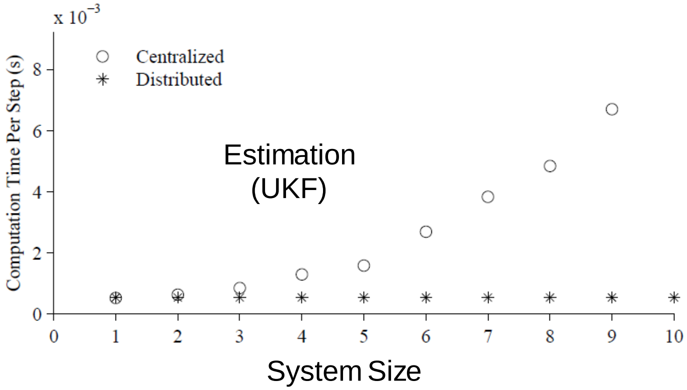
80% Probability of Unsafe Event

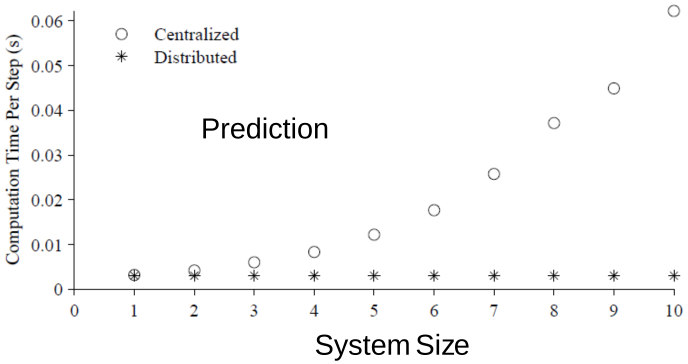
Computational Architecture of RTSM



Distributed Prognostics

- … but the previous algorithms do not scale!





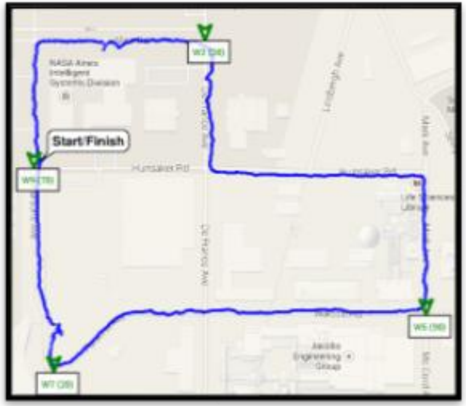
- A distributed solution is needed for large-scale systems, and for system-level prognostics problems
- Propose to *decompose* the *global* prognostics problem, by decomposing the *global model*, into *local* independent subproblems for *local submodels*
  - Assuming some measurements to be known inputs
- Independent subproblems are trivially distributed and parallelized – This is the case for RTSM

Prognostics and Decision-Making

- We employ prognostics in order to inform some type of action
- Autonomous vehicles like UAVs and rovers receive command sequences from humans
  - E.g., as a set of waypoints with scientific objectives to achieve at each
- Unexpected situations can cause the vehicle to go into a safe mode while engineers diagnose the problem, which might take a long time
- An autonomous decision-making system that includes automated diagnosis and prognosis in making optimal decisions can save time, money, and increase mission value

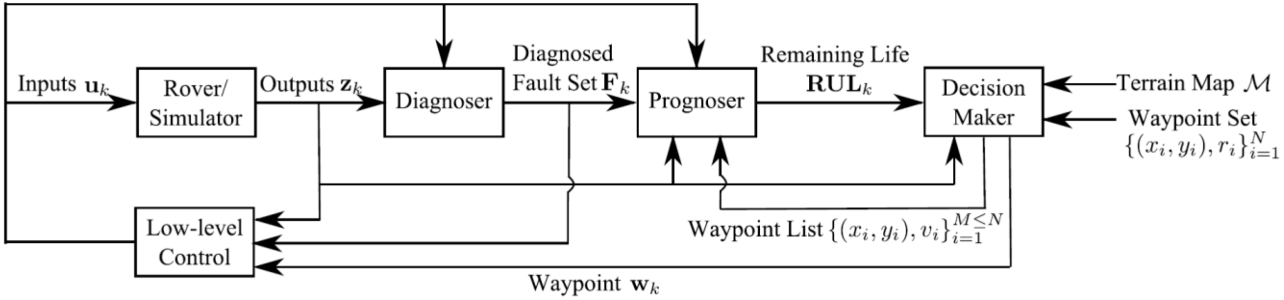
Example: Rover Mission Replanning Using Prognostics

- **Given:**
  - An initial mission route (not necessarily optimized) which includes waypoint parameter constraints
  - Each waypoint is associated with a payoff value
  - A healthy vehicle is able to complete the entire route within the energy and component health constraints
  - A fault occurs that makes it impossible to complete the mission before the End of Life (EoL):
    - **Rover:** parasitic electrical load. The fault results in increased energy consumption and battery overheating.
- **Goal:**
  - Predict end-of-discharge states and replan path that maximizes payoff and extends the remaining useful life



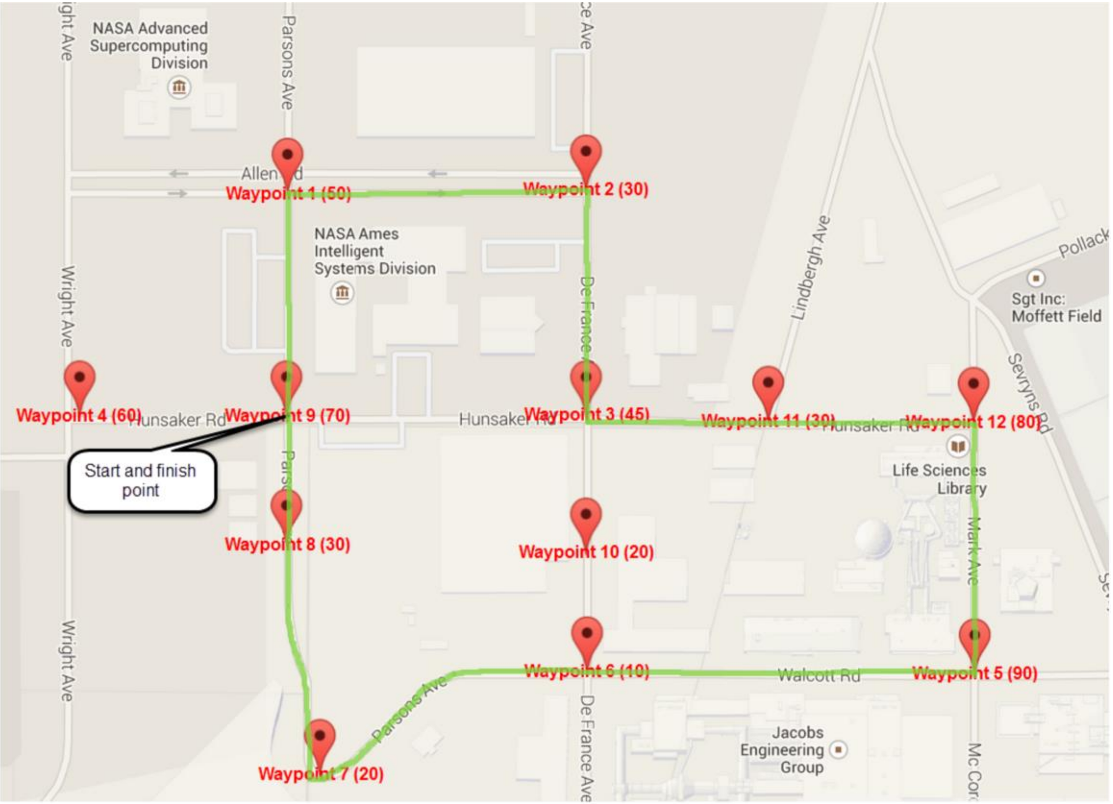
Integrated Decision Making Architecture

1. Rover receives control inputs (individual wheel speeds) and sensors produce outputs
2. Low-level control modifies wheel speed commands to move towards a given waypoint in the presence of diagnosed faults
3. Diagnoser receives rover inputs and outputs and produces fault candidates
4. Prognoser receives rover inputs and outputs and predicts remaining useful life (RUL) or rover and/or its components (eg, batteries, motors)
5. Decision maker plans the order to visit the waypoints (science objectives) given diagnostic and prognostic information. It can also selectively eliminate some of the waypoints if all of them are not achievable due to vehicle health or energy constraints.

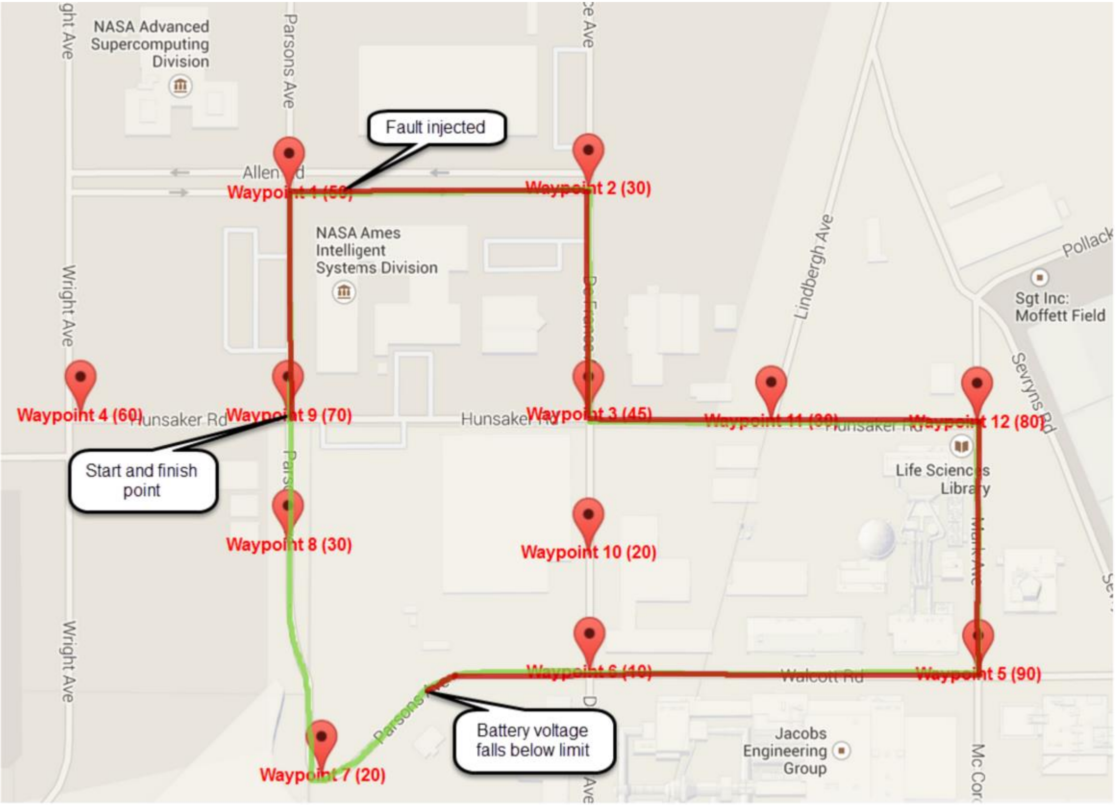


Inputs $\mathbf{u}_k$
Rover/ Simulator
Outputs $\mathbf{z}_k$
Diagnoser
Diagnosed Fault Set $\mathbf{F}_k$
Prognoser
Remaining Life $\mathbf{RUL}_k$
Decision Maker
Terrain Map $\mathcal{M}$
Waypoint Set $\{(x_i, y_i), r_i\}_{i=1}^{N}$
Low-level Control
Waypoint List $\{(x_i, y_i), v_i\}_{i=1}^{M \leq N}$
Waypoint $\mathbf{w}_k$

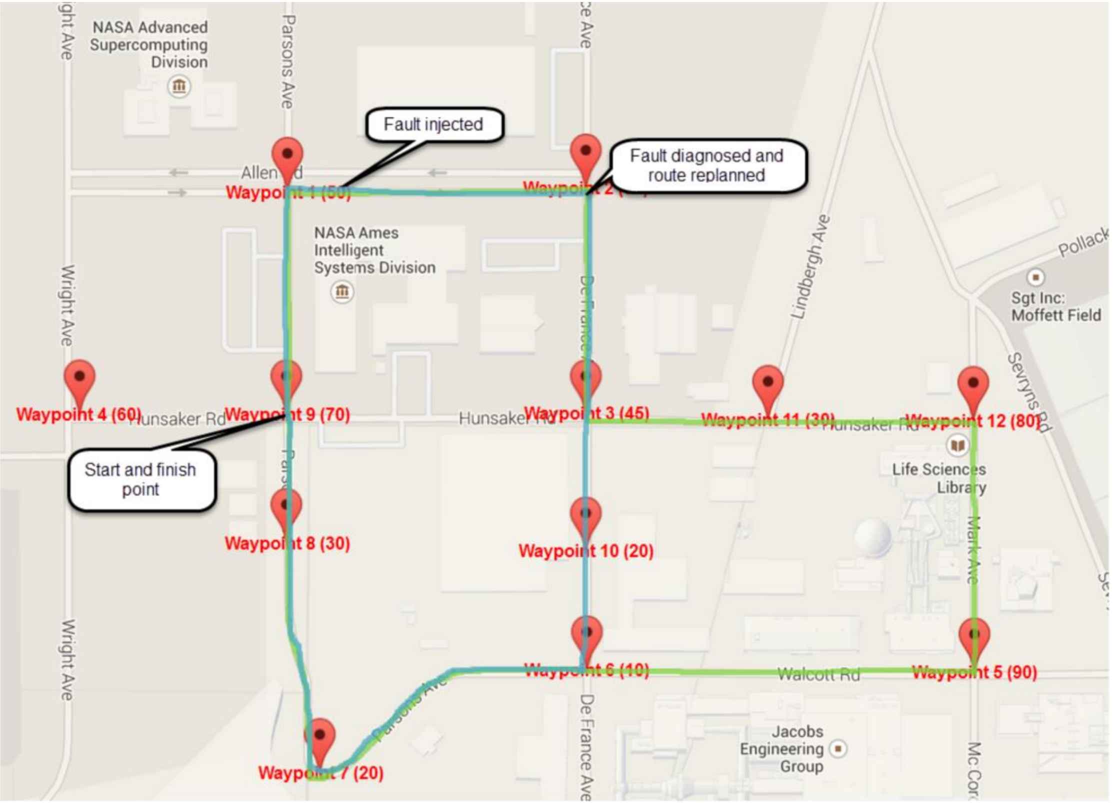
Rover Replanning: Nominal Scenario



Rover Replanning: Fault Injected, PDM Disabled



Rover Replanning: Fault Injected, PDM Enabled



Wrapping Up

Conclusions

- Presented model-based prognostics framework
- Key takeaways:
  - Modeling is key – both dynamics of the system and representation of uncertain inputs to the prediction problem
  - Uncertainty is inherent to the problem and cannot be ignored
  - Future input uncertainty is often most significant and its representation should include as much knowledge about future operation of the system as is known
- Framework and models implemented in open-source MATLAB packages
  - https://github.com/nasa/PrognosticsModelLibrary
  - https://github.com/nasa/PrognosticsAlgorithmLibrary

Acknowledgements

- Diagnostics and Prognostics Group, NASA Ames Research Center
  - Kai Goebel, Scott Poll, Edward Balaban, Chetan Kulkarni, Shankar Sankararaman, Adam Sweet, Lilly Spirkovska, Chris Teubert, George Gorospe, Ann Patterson-Hine
- Former NASA
  - Matthew Daigle, Abhinav Saxena, Jose Celaya, Sriram Narasimhan
- University of Valladolid, Spain
  - Anibal Bregon
- Some of the slides have been adapted from Matthew Daigle's presentation slides on related topics

Selected Bibliography

- K. Goebel, M. Daigle, A. Saxena, S. Sankararaman, I. Roychoudhury, and J. Celaya, "Prognostics: The Science of Making Predictions," April 2017. (Available on Amazon http://a.co/0xXzmJ6)
- M. Daigle, A. Bregon, and I. Roychoudhury, "Distributed Prognostics Based on Structural Model Decomposition," IEEE Transactions on Reliability, vol. 63, no. 2, pp. 495-510, June 2014.
- S. Sankararaman, M. Daigle, and K. Goebel, "Uncertainty Quantification in Remaining Useful Life Prediction using First-Order Reliability Methods," IEEE Transactions on Reliability, vol. 63, no. 2, pp. 603-619, June 2014.
- M. Daigle and C. Kulkarni, "Electrochemistry-based Battery Modeling for Prognostics," Annual Conference of the Prognostics and Health Management Society 2013, pp. 249-261, October 2013.pace Conference, March 2014.
- M. Daigle and S. Sankararaman, "Advanced Methods for Determining Prediction Uncertainty in Model-Based Prognostics with Application to Planetary Rovers," Annual Conference of the Prognostics and Health Management Society 2013, pp. 262-274, October 2013.
- E. Balaban, S. Narasimhan, M. Daigle, I. Roychoudhury, A. Sweet, C. Bond, and G. Gorospe, "Development of a Mobile Robot Test Platform and Methods for Validation of Prognostics-Enabled Decision Making Algorithms," International Journal of Prognostics and Health Management, vol. 4, no. 1, May 2013.
- M. Daigle and K. Goebel, "Model-based Prognostics with Concurrent Damage Progression Processes," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 43, no. 4, pp. 535-546, May 2013.
- I. Roychoudhury, M. Daigle, A. Bregon, and B. Pulido, "A Structural Model Decomposition Framework for Systems Health Management," Proceedings of the 2013 IEEE Aerospace Conference, March 2013.
- M. Daigle, A. Bregon, and I. Roychoudhury, "A Distributed Approach to System-Level Prognostics," Annual Conference of the Prognostics and Health Management Society 2012, pp. 71-82, September 2012.
- A. Bregon, M. Daigle, and I. Roychoudhury, "An Integrated Framework for Model-based Distributed Diagnosis and Prognosis," Annual Conference of the Prognostics and Health Management Society 2012, pp. 416-426, September 2012.
- I. Roychoudhury and M. Daigle, "An Integrated Model-Based Diagnostic and Prognostic Framework," Proceedings of the 22nd International Workshop on Principles of Diagnosis, pp. 44-51, October 2011.
- M. Daigle, I. Roychoudhury, S. Narasimhan, S. Saha, B. Saha, and K. Goebel, "Investigating the Effect of Damage Progression Model Choice on Prognostics Performance," Proceedings of the Annual Conference of the Prognostics and Health Management Society 2011, pp. 323-333, September 2011.
- M. Daigle and K. Goebel, "A Model-based Prognostics Approach Applied to Pneumatic Valves," International Journal of Prognostics and Health Management, vol. 2, no. 2, August 2011.
- Orchard, M., Cerda, M., Olivares, B., and Silva, J., "Sequential Monte Carlo Methods for Discharge Time Prognosis in Lithium-Ion Batteries," International Journal of Prognostics and Health Management, Vol. 3, Issue 2 (010), pp. 1-12, 2012.
- Orchard, M., Tobar, F., and Vachtsevanos, G., "Outer Feedback Correction Loops in Particle Filtering-based Prognostic Algorithms: Statistical Performance Comparison," Studies in Informatics and Control, vol. 18, Issue 4, pp. 295-304, December 2009.
- Orchard, M. and Vachtsevanos, G., "A Particle Filtering Approach for On-Line Fault Diagnosis and Failure Prognosis," Transactions of the Institute of Measurement and Control, vol. 31, no. 3-4, pp. 221-246, June 2009.
- B. Saha and K. Goebel, "Modeling Li-ion battery capacity depletion in a particle filtering framework," in Proceedings of the Annual Conference of the Prognostics and Health Management Society 2009, Sept. 2009.
- J. Luo, K. R. Pattipati, L. Qiao, and S. Chigusa, "Model-based prognostic techniques applied to a suspension system," IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans, vol. 38, no. 5, pp. 1156 – 1168, Sept. 2008.
- B. Saha and K. Goebel, "Model adaptation for prognostics in a particle filtering framework," International Journal of Prognostics andHealth Management, vol. 2, no. 1, 2011.
- A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, "Metrics for offline evaluation of prognostic performance," International Journal of Prognostics and Health Management, vol. 1, no. 1, 2010.
- E. Zio and G. Peloni, "Particle filtering prognostic estimation of the remaining useful life of nonlinear components," Reliability Engineering & System Safety, vol. 96, no. 3, pp. 403–409, 2011.
- M. Roemer, C. Byington, G. Kacprzynski, and G. Vachtsevanos, "An overview of selected prognostic technologies with reference to an integrated PHM architecture," in Proceedings of the First International Forum on Integrated System Health Engineering and Management in Aerospace, 2005.
- C. S. Kulkarni, J. R. Celaya, G. Biswas, and K. Goebel, "Physics Based Degradation Models for Capacitor Prognostics under Thermal Overtress Conditions,“ International Journal of Prognostics and Health Management, Vol. 4 No. 1, 2013.

Useful Links

- **PCOE Webpage**: http://prognostics.nasa.gov
- **PCOE Data repository**: https://ti.arc.nasa.gov/tech/dash/pcoe/prognostic-data-repository/
- **Generic Software Architecture for Prognostics (GSAP)** https://github.com/nasa/GSAP

indranil.roychoudhury@nasa.gov

http://prognostics.nasa.gov