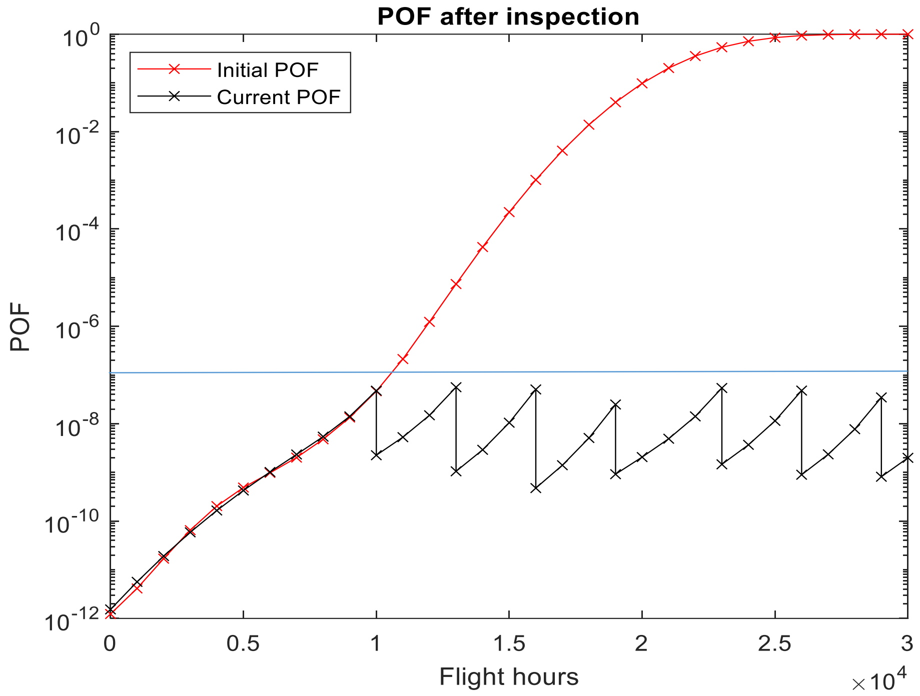
Risk Based Optimized Inspections for Aircraft Fleets

Juan D. Ocampo
and Daniela Posso-Hoyos
St. Mary's University
Harry Millwater
University of Texas at San Antonio

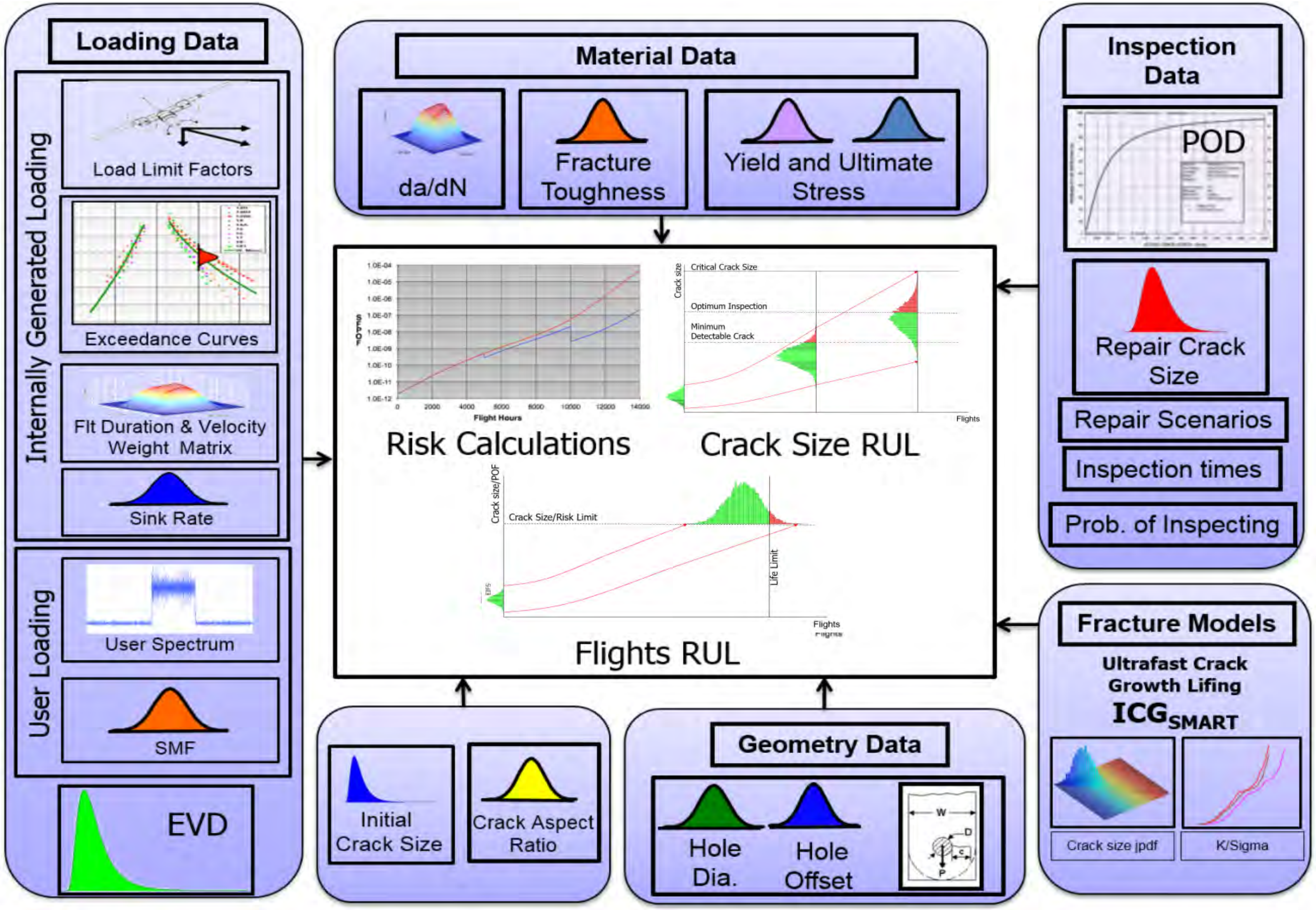
19th International Workshop on the Holistic Structural Integrity Process (HOLSIP) - Salt Lake City, UT - February 12 2020.

Outline

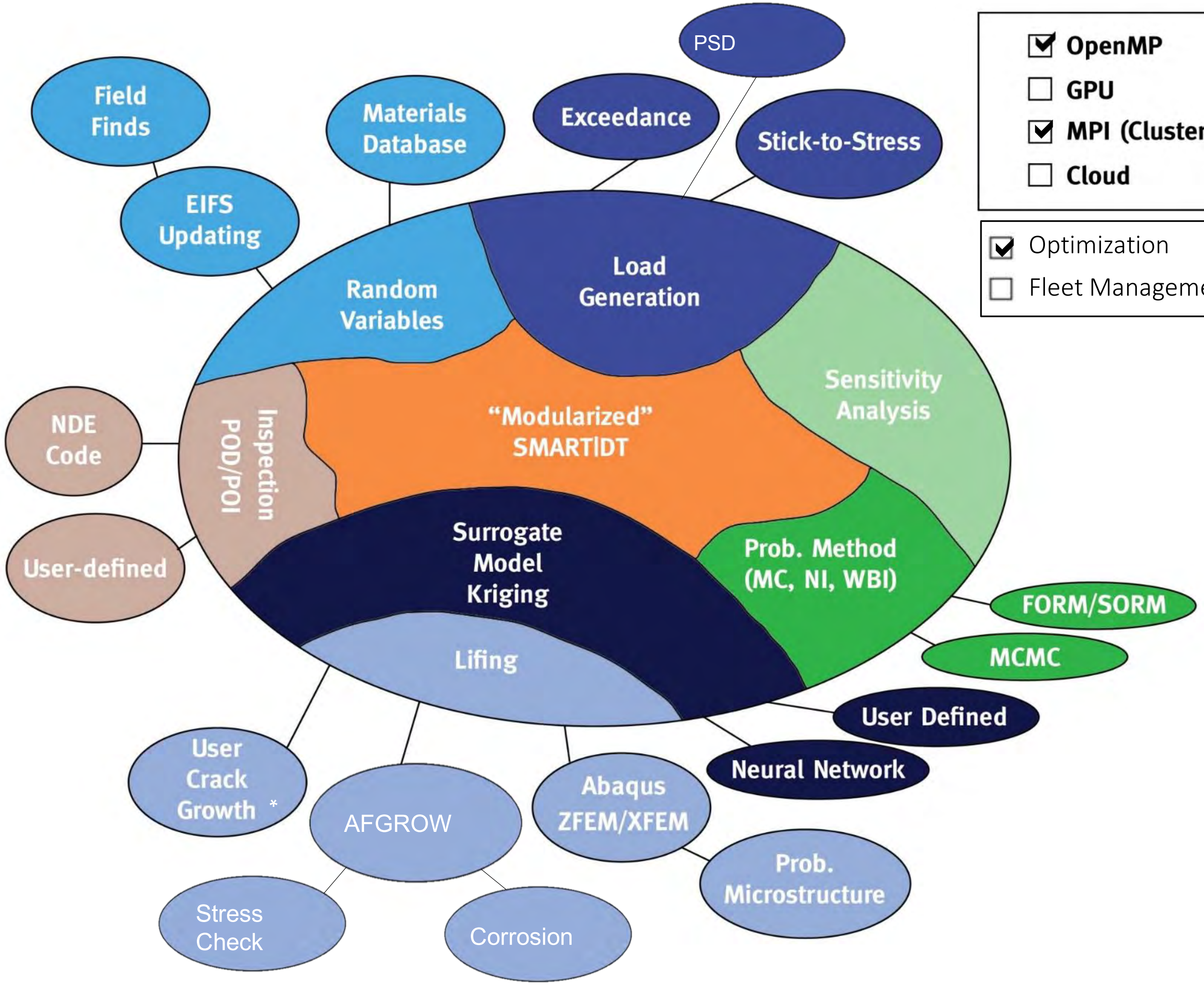
- ✓Probabilistic Damage Tolerance Analysis Quick Review
- ✓ Optimized Risk Inspections
  - ✓Risk Threshold Method
  - ✓Shortest Path Method
    - ✓Single Inspection
    - ✓Multiple inspections and Cost Minimization
    - ✓Skipping Algorithm
  - ✓Example Problem
- ✓Future Plans
- ✓Conclusions



SMART


Loading Data
Internally Generated Loading
Load Limit Factors
Exceedance Curves
Flt Duration & Velocity Weight Matrix
Sink Rate
User Loading
User Spectrum
SMF
EVD
Material Data
da/dN
Fracture Toughness
Yield and Ultimate Stress
Risk Calculations
Flight Hours
Crack Size RUL
Critical Crack Size
Optimum Inspection
Minimum Detectable Crack
Flights
Crack Size/Risk Limit
Life Limit
Flights RUL
Inspection Data
POD
Repair Crack Size
Repair Scenarios
Inspection times
Prob. of Inspecting
Fracture Models
Ultrafast Crack Growth Lifing
ICG SMART
Crack size jpdf
K/Sigma
Initial Crack Size
Crack Aspect Ratio
Geometry Data
Hole Dia.
Hole Offset

The Holistic View of SMART



- [x] OpenMP
- [ ] GPU
- [x] MPI (Clusters)
- [ ] Cloud

- [x] Optimization
- [ ] Fleet Management

Motivation

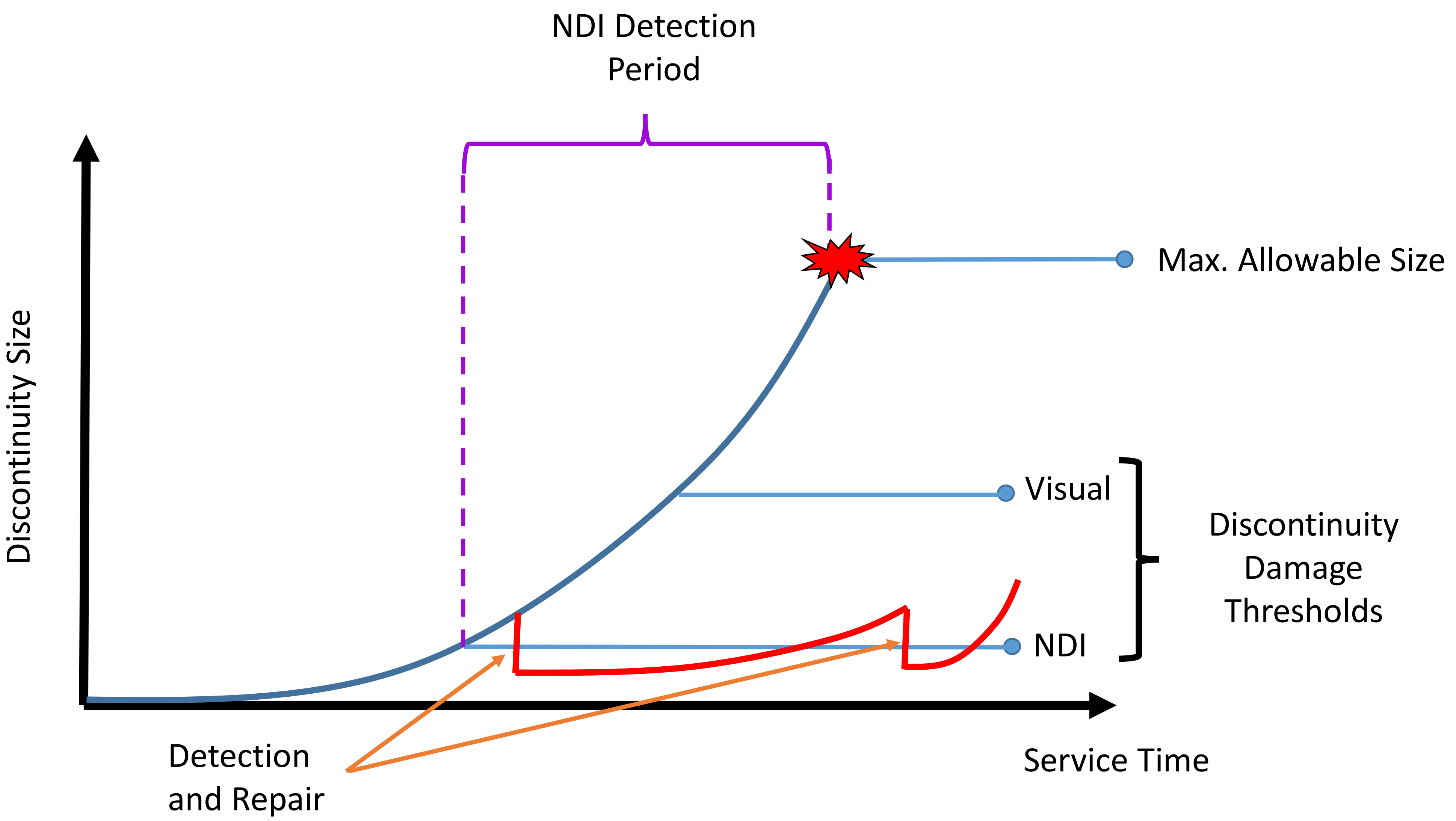
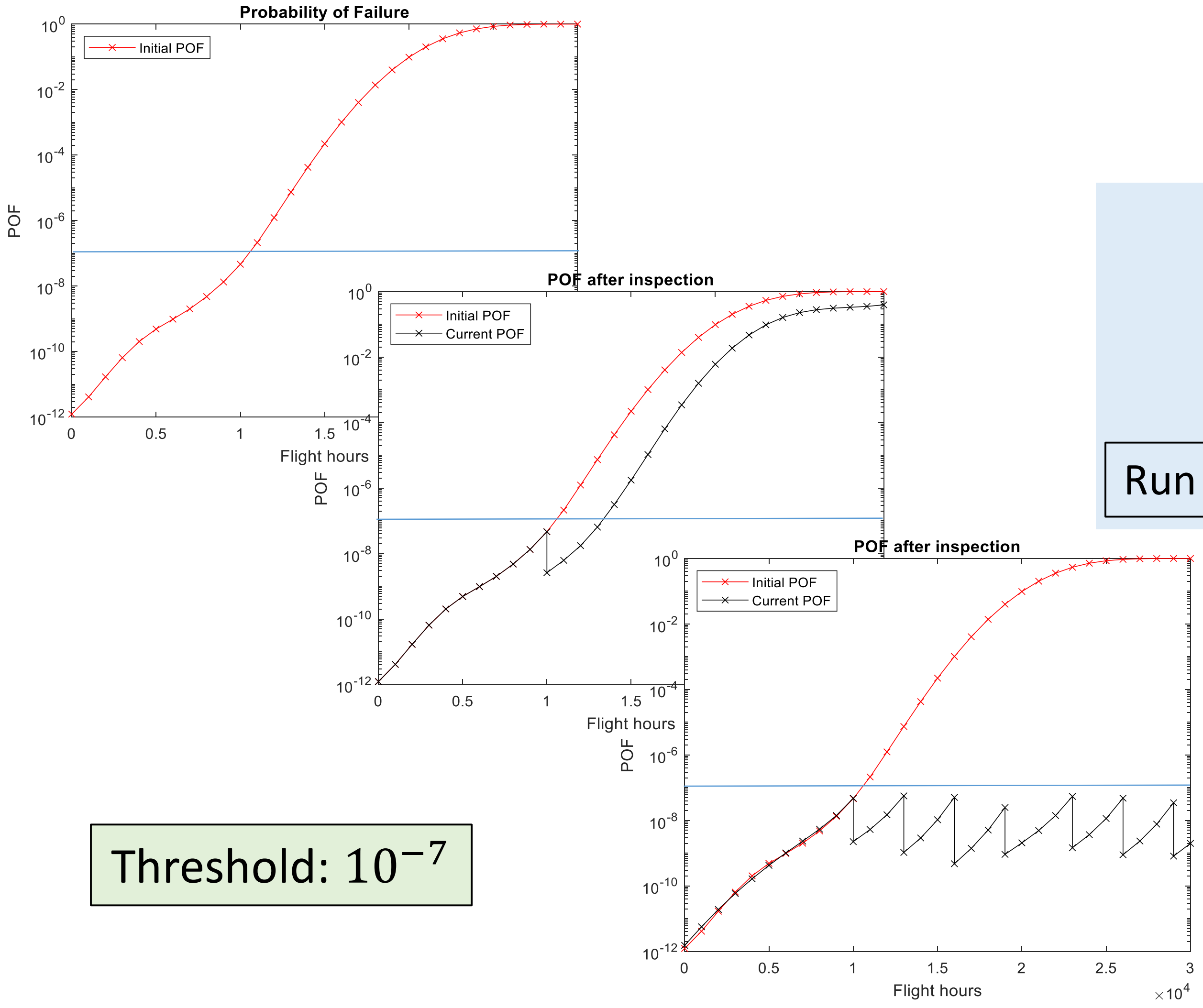
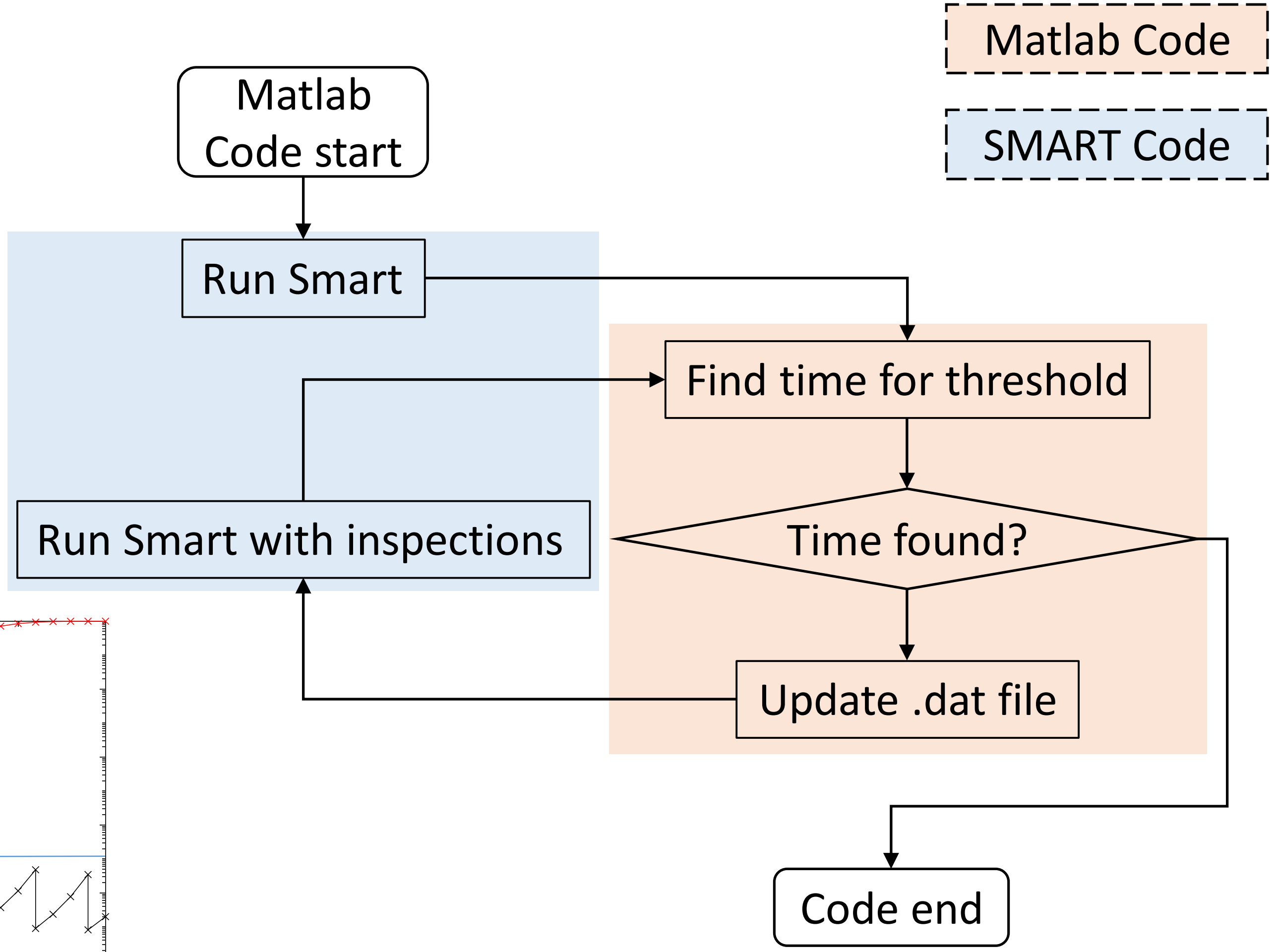

NDI Detection
Period
Max. Allowable Size
Visual
NDI
Discontinuity
Damage
Thresholds
Discontinuity Size
Service Time
Detection
and Repair

Table of capabilities

	Keep risk threshold method	Shortest path method
Operates under a risk threshold constraint	●	●
Inspection times are arbitrarily selected depending on time resolution indicated in SMART	●	●
Inspection times are selected from user defined candidates inspection times		●
Performance with different types of inspections		●
Cost information set per type of inspection thru time is taking into account		●

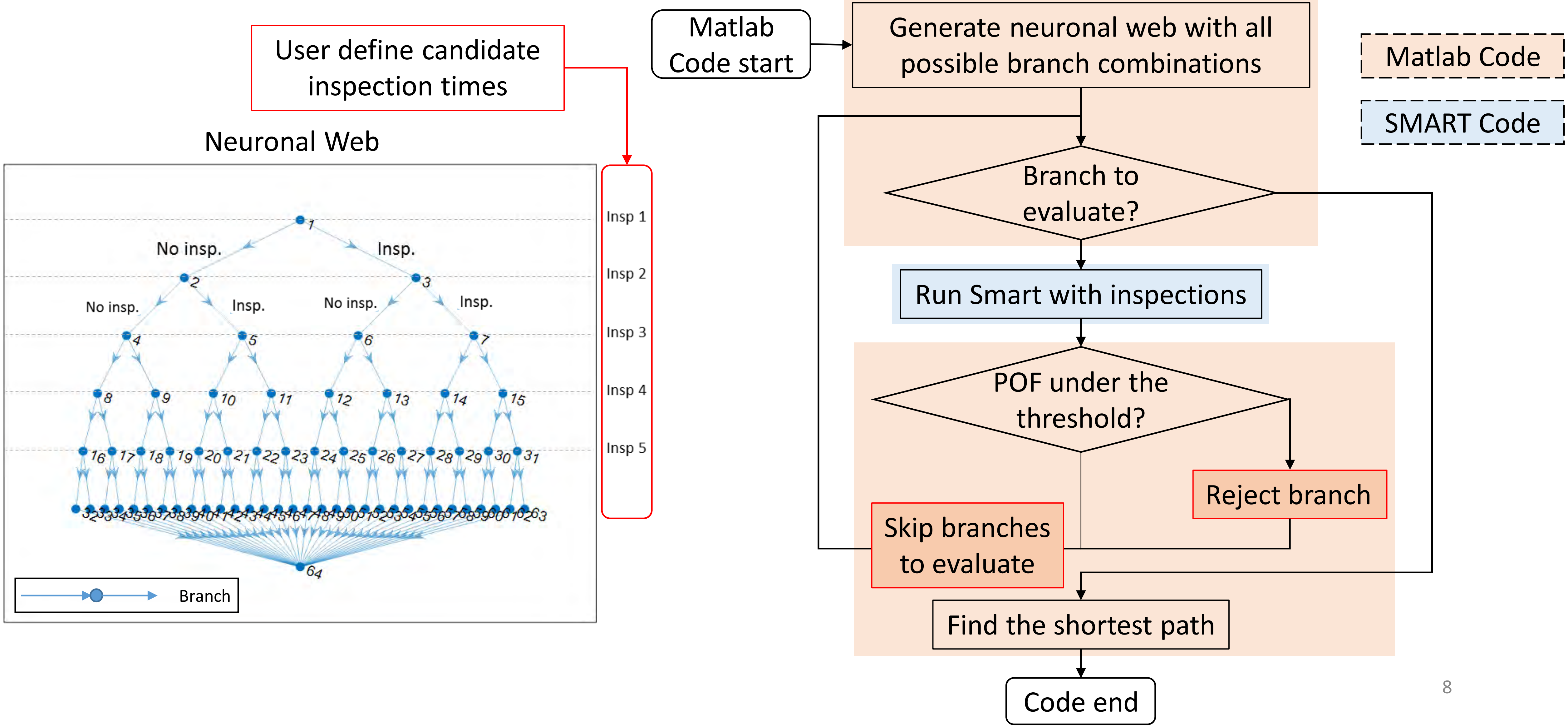
Keep risk threshold





Threshold: 10^{-7}

Shortest Path Method



Shortest path formulation

The decision tree G(V,A) is described by the set of vertices V and its corresponding set of arcs A.

$$C = \{c_{ij} \ / \ c_{ij} \ is \ the \ cost \ of \ traversing \ the \ link \ between \ i \ and \ j\}$$

$$X = \{x_{ij} \ / \ x_{ij} \ \text{ is } 1 \ for \ the \ decision \ of \ travel \ through \ the \ link \ (i,j) \ and \ 0 \ otherwise\}$$

$$V = \{Set \ of \ vertices \ of \ the \ graph\}$$

$$A = \{Set \ of \ arcs \ of \ the \ graph\}$$

Minimize

$$\sum_{(i,j)\in A} C_{ij} X_{ij}$$

subject to

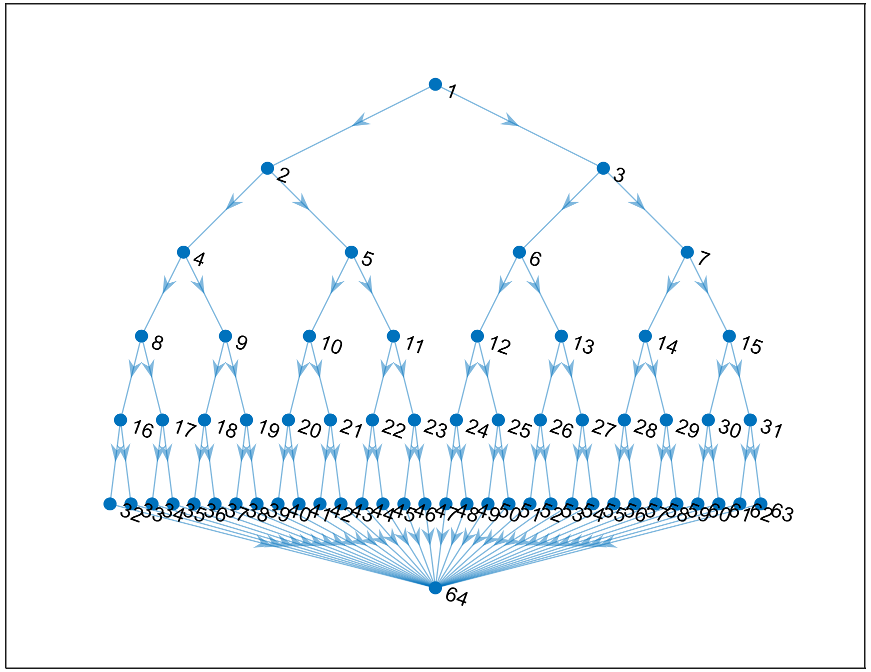
If $\quad F_k > \ threshold$

then $\quad X_{kj} = 0, \ \forall \ j$

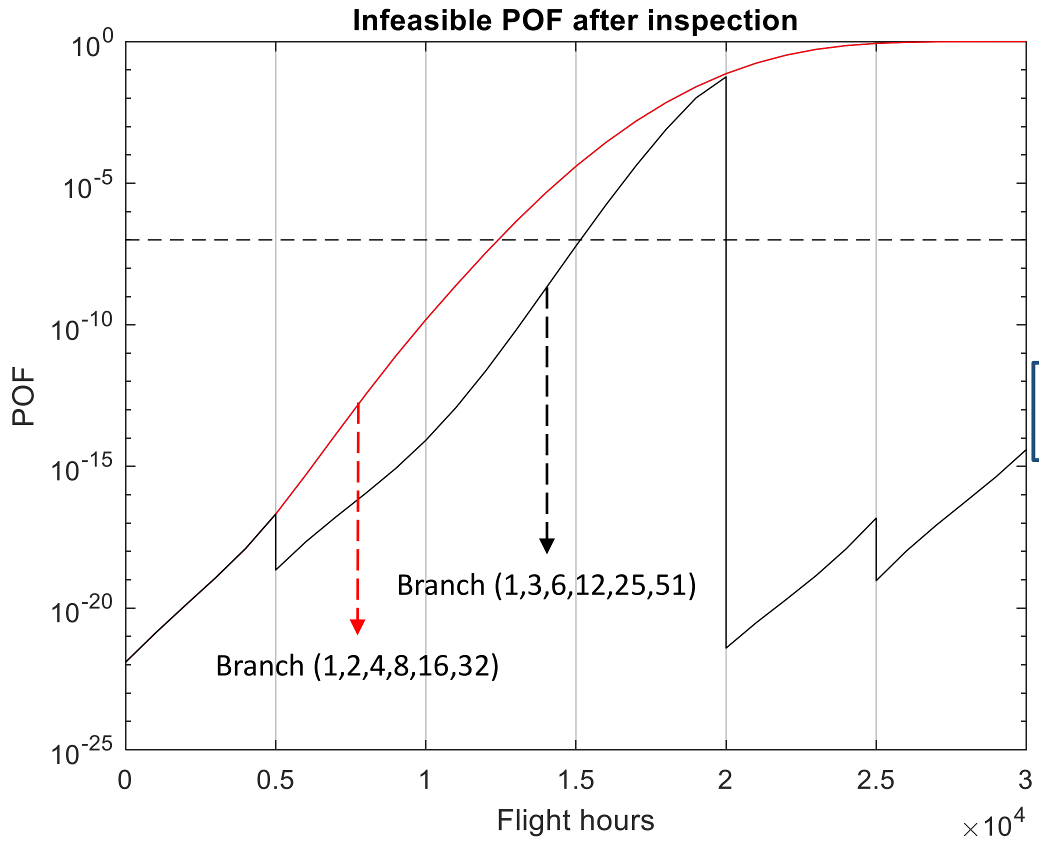
$$C_{ij} = M$$

$$X_{ij} \leq F_i, \forall \ j$$

$$x_{ij} \ \in \ \{0,1\}$$

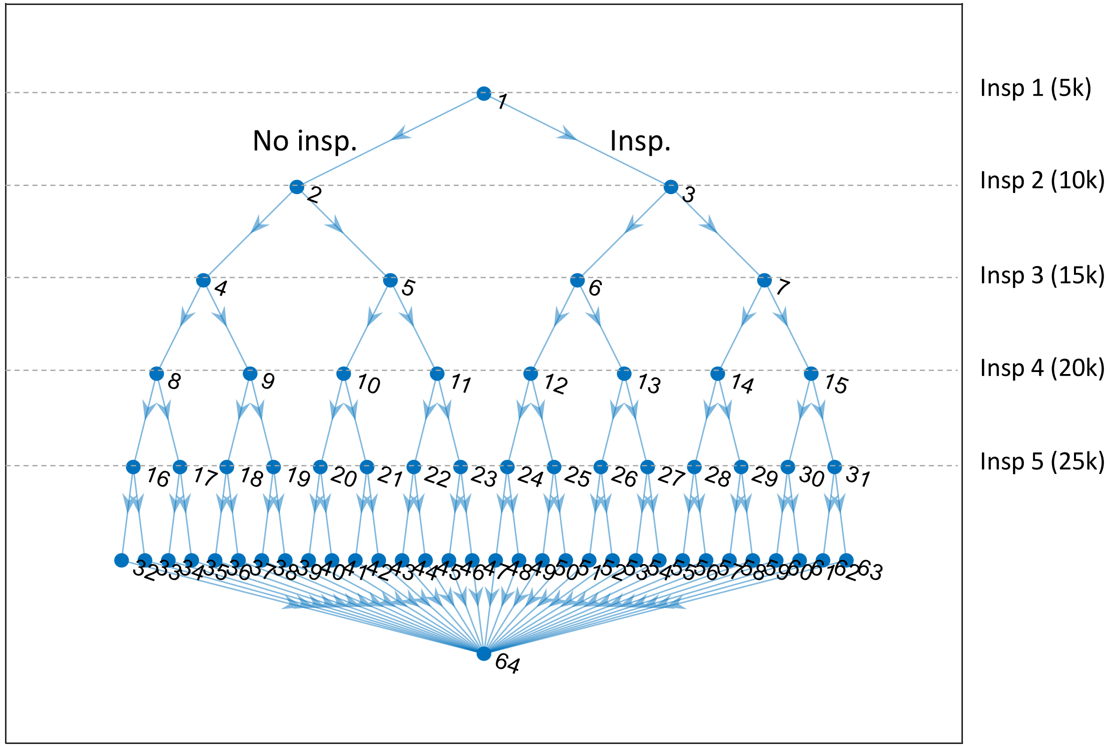


Shortest Path - Single Inspection



Visualization example

User defined inspections at: 5k, 10k, 15k, 20k, and 25k

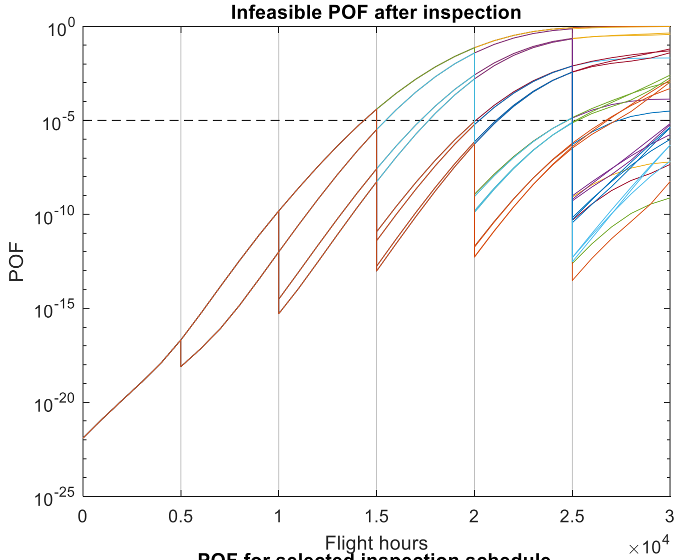


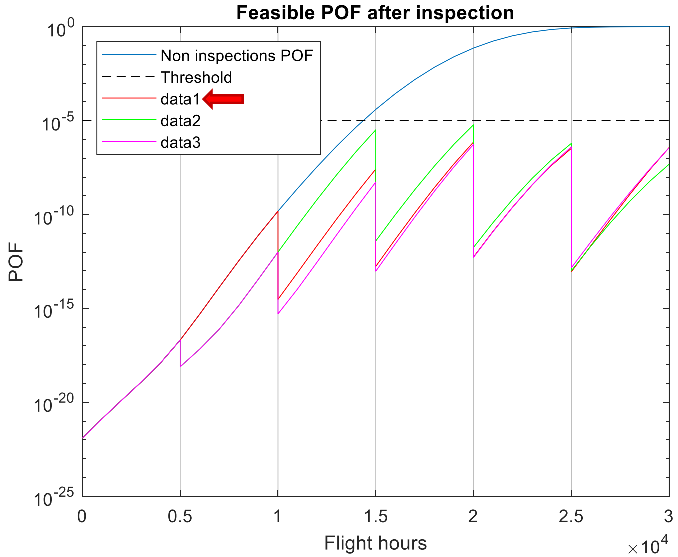
POFs for each branch / inspection schedule

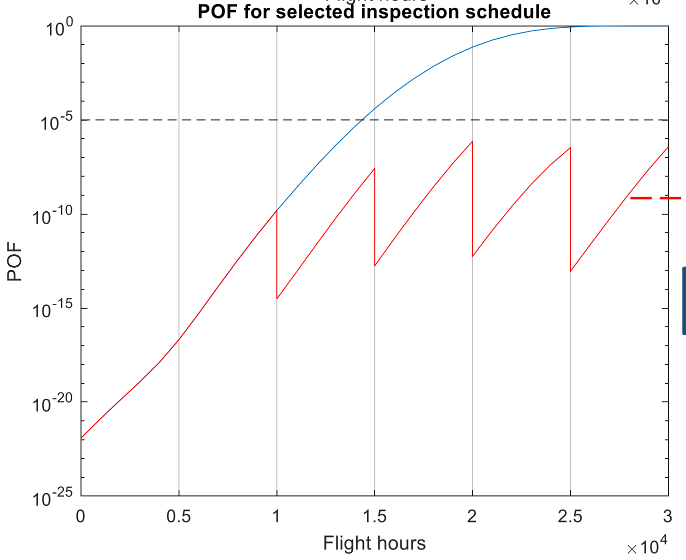
32 SMART Runs

Threshold: 10^{-5}

Shortest Path - Single Inspection



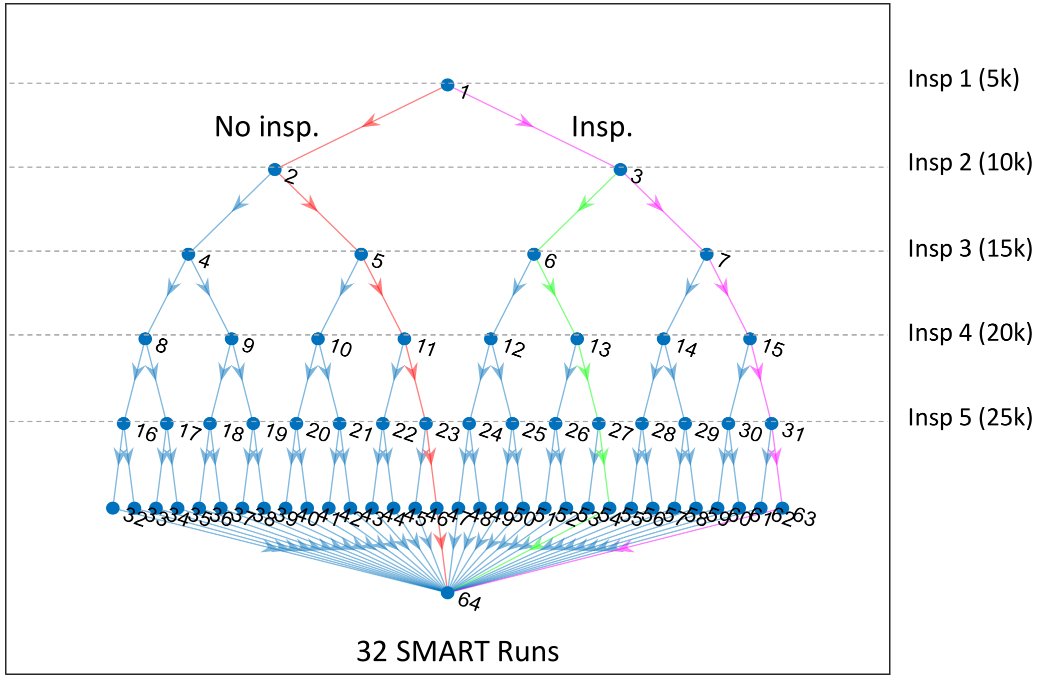




Selected branch
(1,2,5,11,23,47)

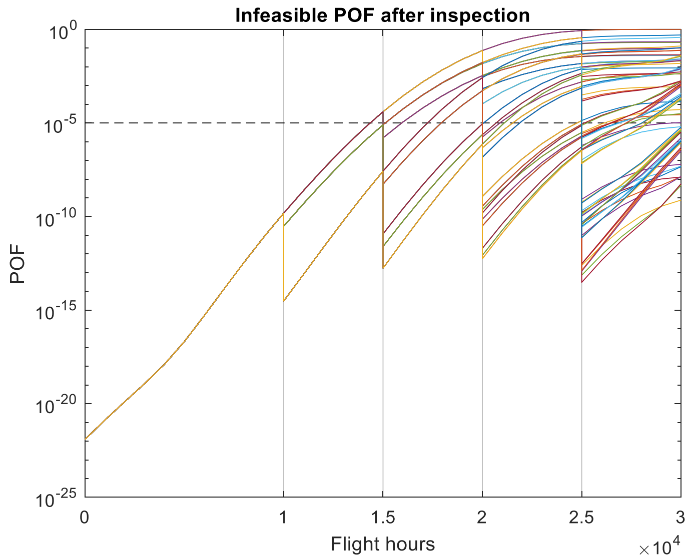
Selected schedule [10000,15000,20000,25000]

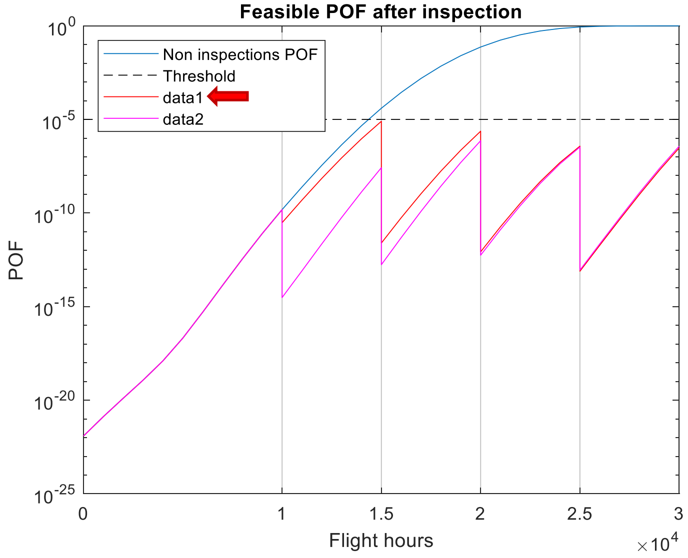
User defined inspections at: 5k, 10k, 15k, 20k, and 25k



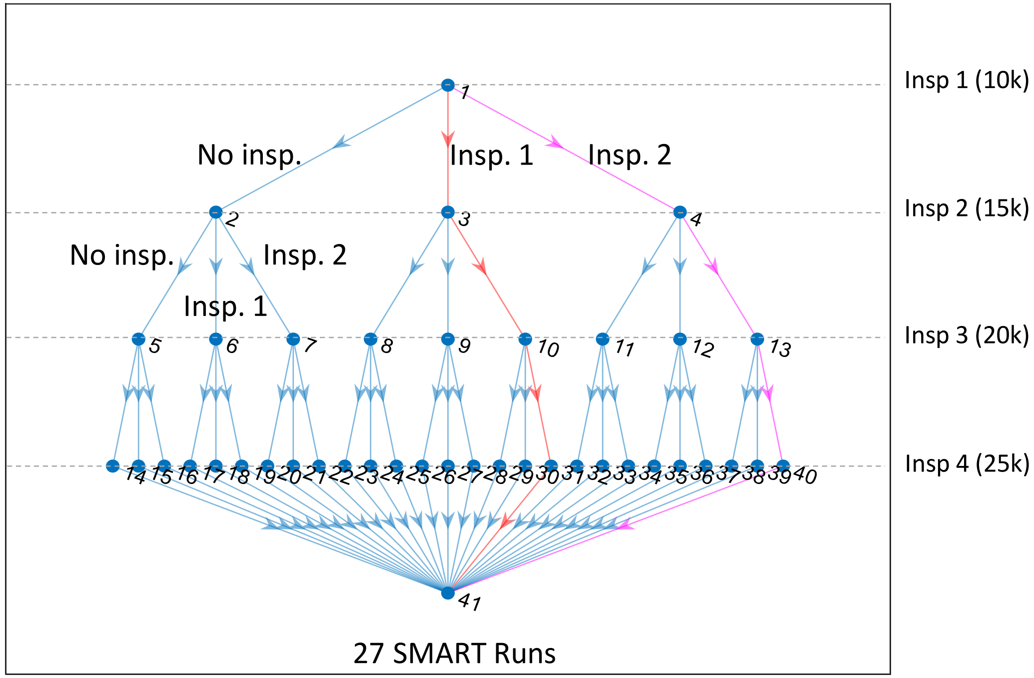
Possible inspection times [5000:5000:25000]

Shortest Path - Single Inspection

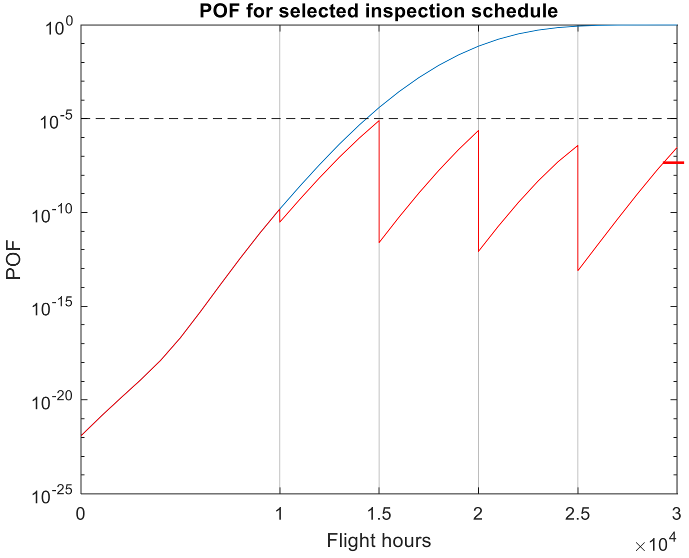




User defined inspections at: 10k, 15k, 20k, and 25k



Possible inspection times [10000,15000,20000,25000]



Selected branch
(1,3,10,31)

Selected schedule [10000,15000,25000]
Selected inspection type [2,2,2]

Inspection and repair costs can be variable thru time

$$ Inspection 1 [	$0.3	$0.3	$0.3	$0.3]
$$ Inspection 2 [	$0.8	$0.8	$0.8	$0.8]

Shortest path with branch skipping algorithm -
User defined inspections

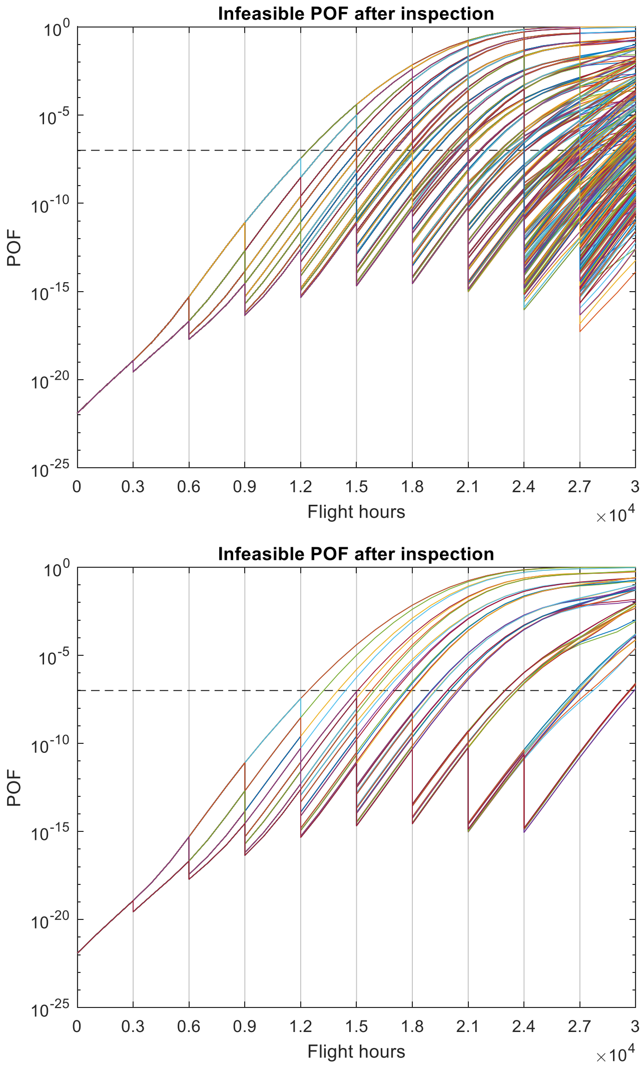
Branches skipping algorithm

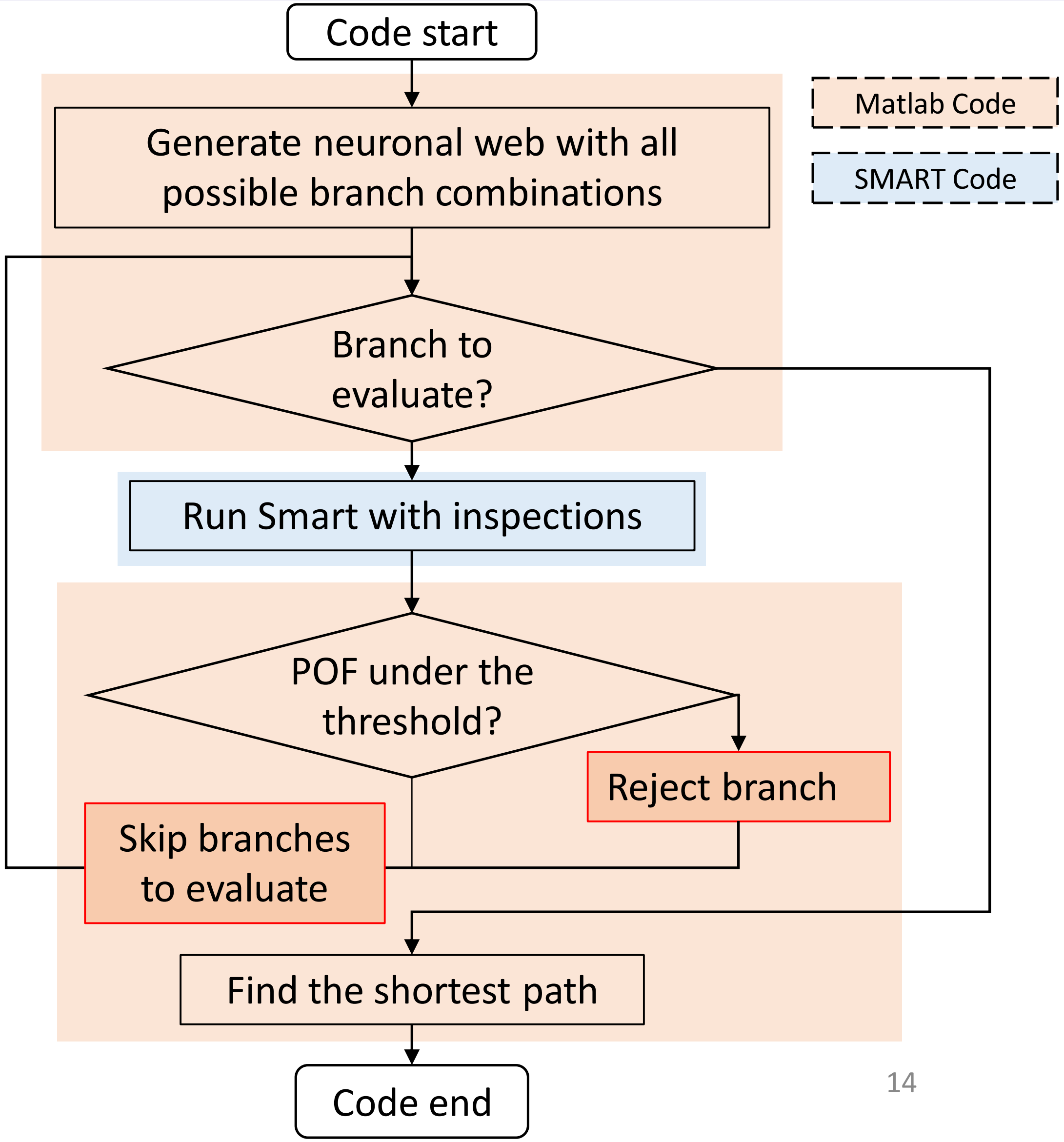
All branches evaluated
512
Without skipping algorithm

Skipping algorithm filters up all possible scenarios

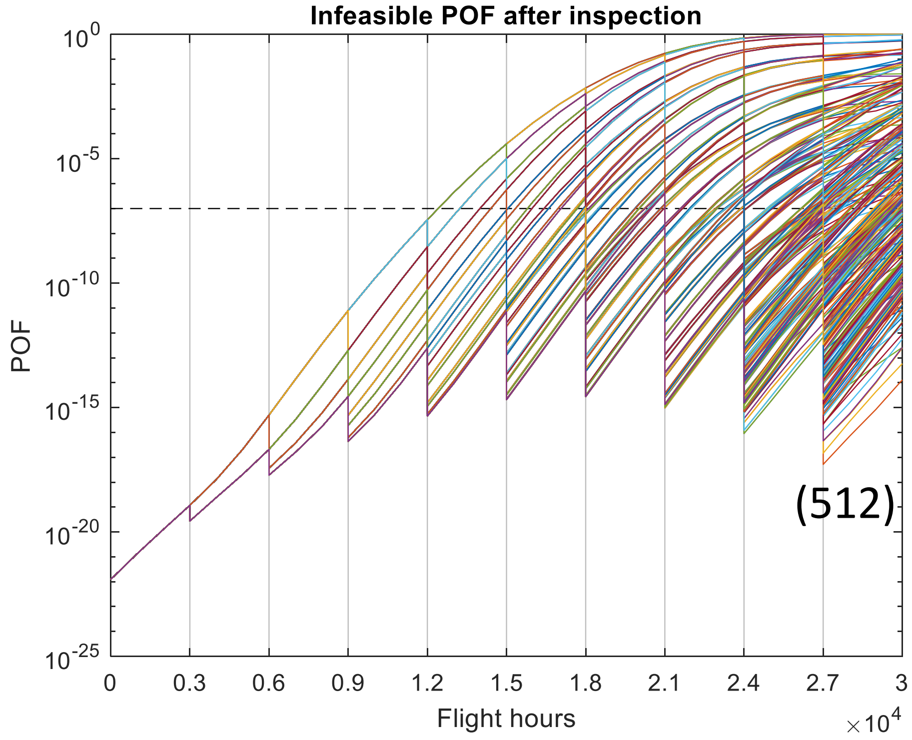
Necessary branches to evaluate
76
With skipping algorithm

Threshold: 10^{-7}





Inspection Combination Matrix
One Inspection Type



Possible inspection times [3000:3000:27000]

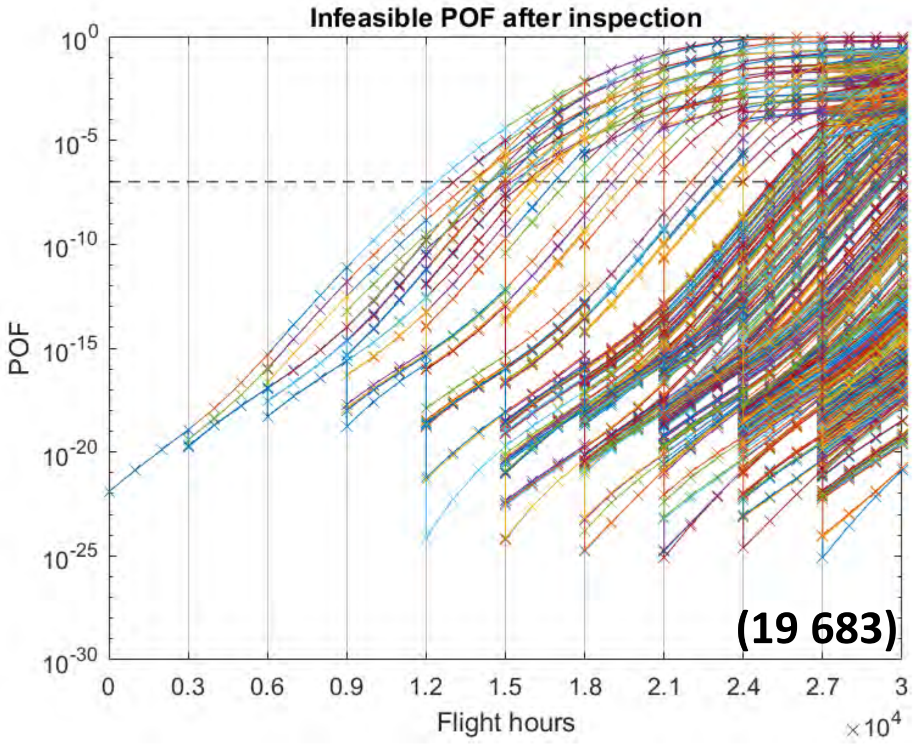
Schedule times (10^3 flight hours)

Schedule combinations	3 2^8	6 2^7	9 2^6	12 2^5	15 2^4	18 2^3	21 2^2	24 2^1	27 2^0
(1)	0	0	0	0	0	0	0	0	0
(2)	0	0	0	0	0	0	0	0	1
(3)	0	0	0	0	0	0	0	1	0
(4)	0	0	0	0	0	0	0	1	1
(5)	0	0	0	0	0	0	1	0	0
	:	:	:	:	:	:	:	:	:
(i)	**Binary(i - 1)**								
	:	:	:	:	:	:	:	:	:
(512)	1	1	1	1	1	1	1	1	1

Number of possible inspections times: number of positions that will be fill with all the numerical combinations in base 2

One inspection type → *"Inspection or no inspection"* → **Base 2** numbers

Inspection Combination matrix
Two Inspections Types



Possible inspection times [3000:3000:27000]

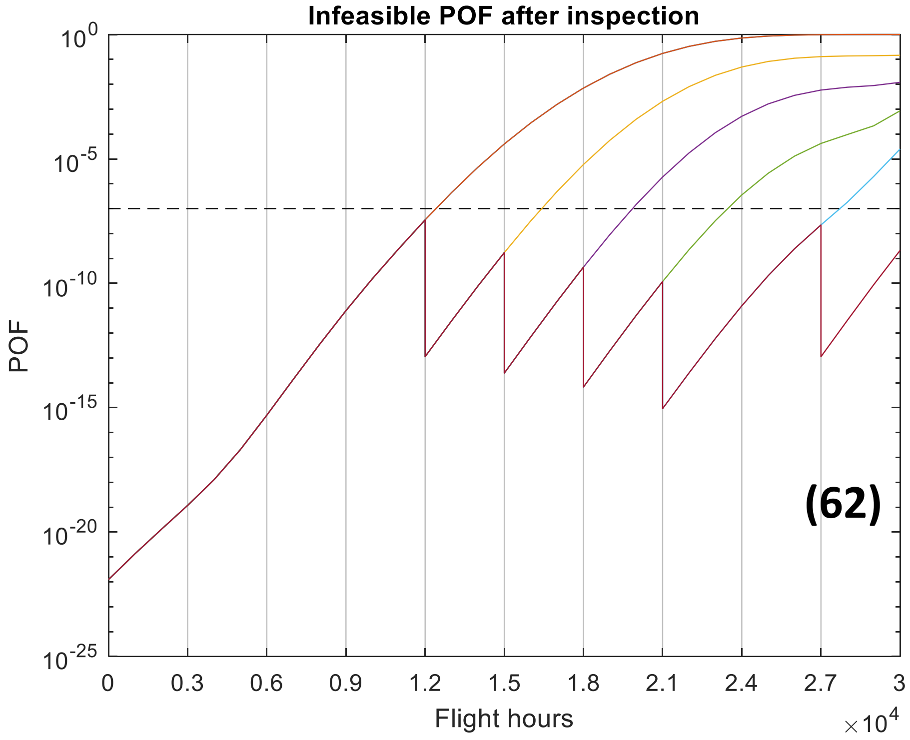
Schedule times (10^3 flight hours)

Schedule combinations	3 (3^8)	6 (3^7)	9 (3^6)	12 (3^5)	15 (3^4)	18 (3^3)	21 (3^2)	24 (3^1)	27 (3^0)
(1)	0	0	0	0	0	0	0	0	0
(2)	0	0	0	0	0	0	0	0	1
(3)	0	0	0	0	0	0	0	0	2
(4)	0	0	0	0	0	0	0	1	0
(5)	0	0	0	0	0	0	0	1	1
(6)	0	0	0	0	0	0	0	1	2
(7)	0	0	0	0	0	0	0	2	0
	:	:	:	:	:	:	:	:	:
(143)	0	0	0	0	1	2	0	2	1
(i)	**Base3(i - 1)**								
(19 683)	2	2	2	2	2	2	2	2	2

Number of possible inspections times: number of positions that will be fill with all the numerical combinations in base 3

Two inspection types → *"Insp. type 1, insp. type 2 or no insp."* → **Base 3** numbers

Reject and Skip Branches Evaluation One Inspection Type



Possible inspection times [3000:3000:27000]

First, POF without inspections

Schedule times (10^3 flight hours)

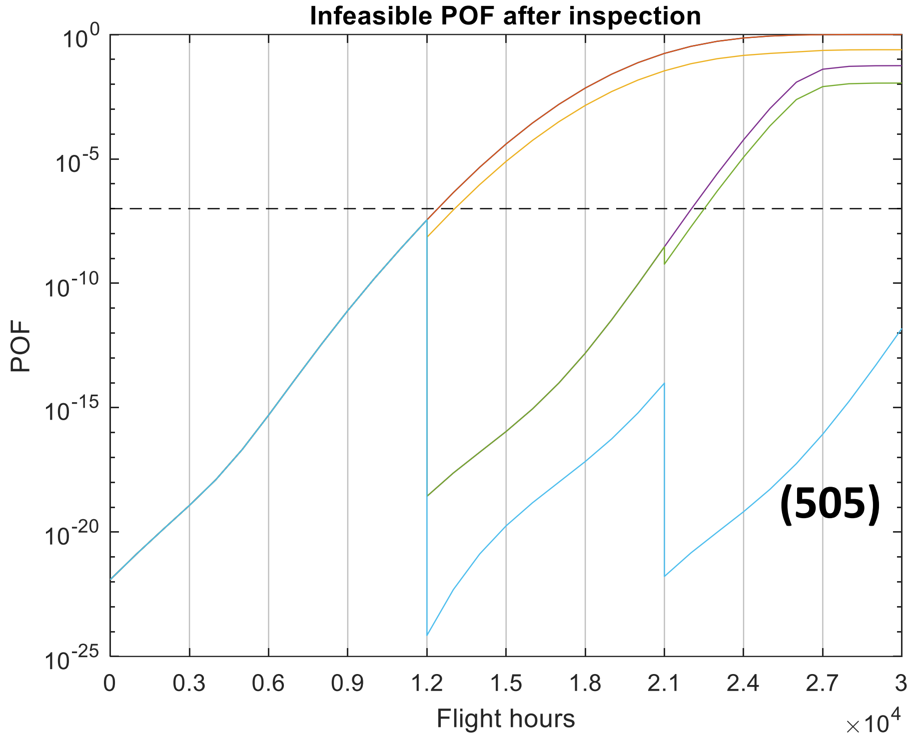
Schedule combinations	3	6	9	12	15	18	21	24	27
(1)	0	0	0	0	0	0	0	0	0
(33)	0	0	0	1	0	0	0	0	0
(49)	0	0	0	1	1	0	0	0	0
(57)	0	0	0	1	1	1	0	0	0
(61)	0	0	0	1	1	1	1	0	0
(62)	0	0	0	1	1	1	1	0	1

+1

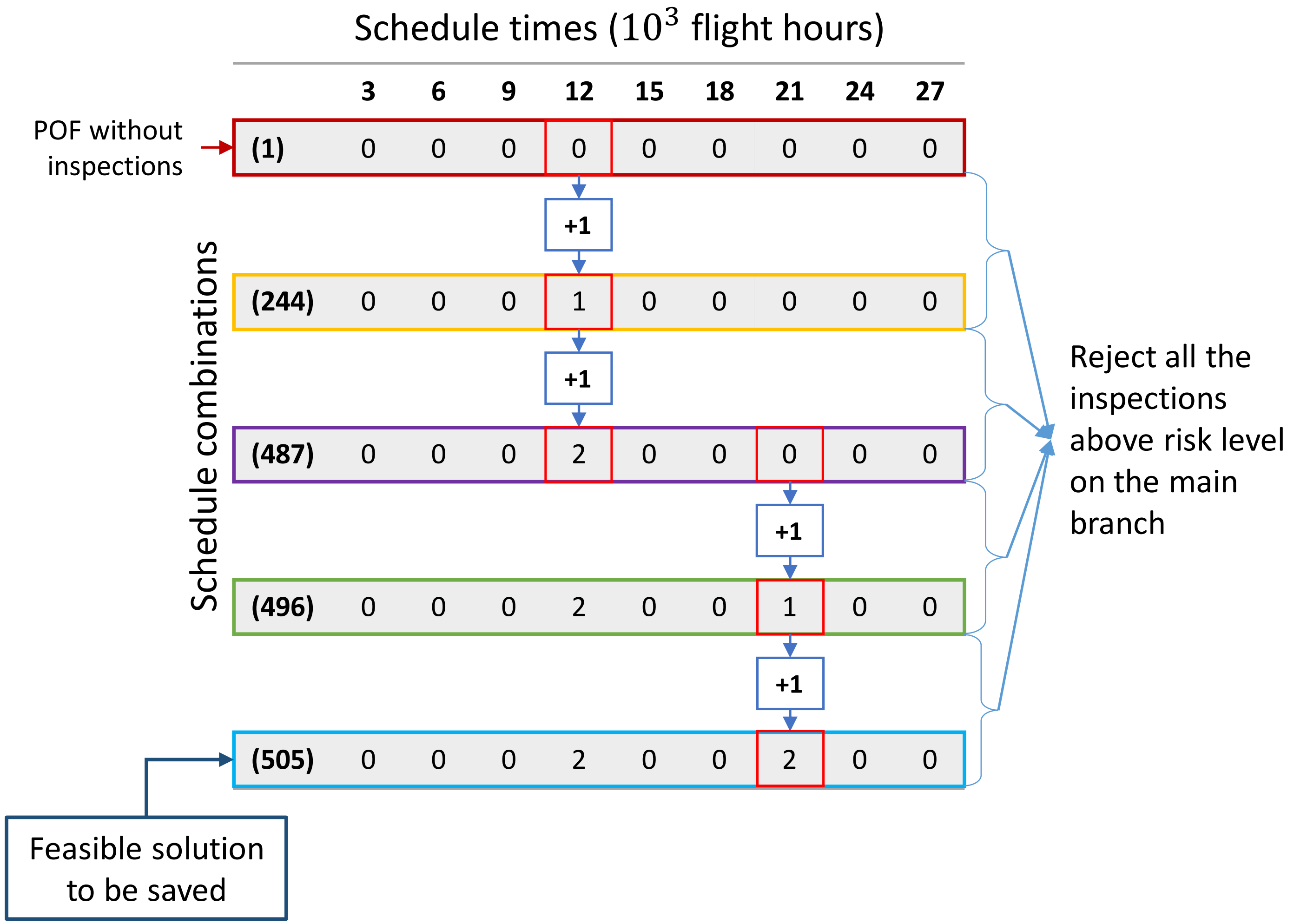
Reject all the inspections above risk level on the main branch

The code evaluates each new combination until get a **feasible solution**, which will be **saved**

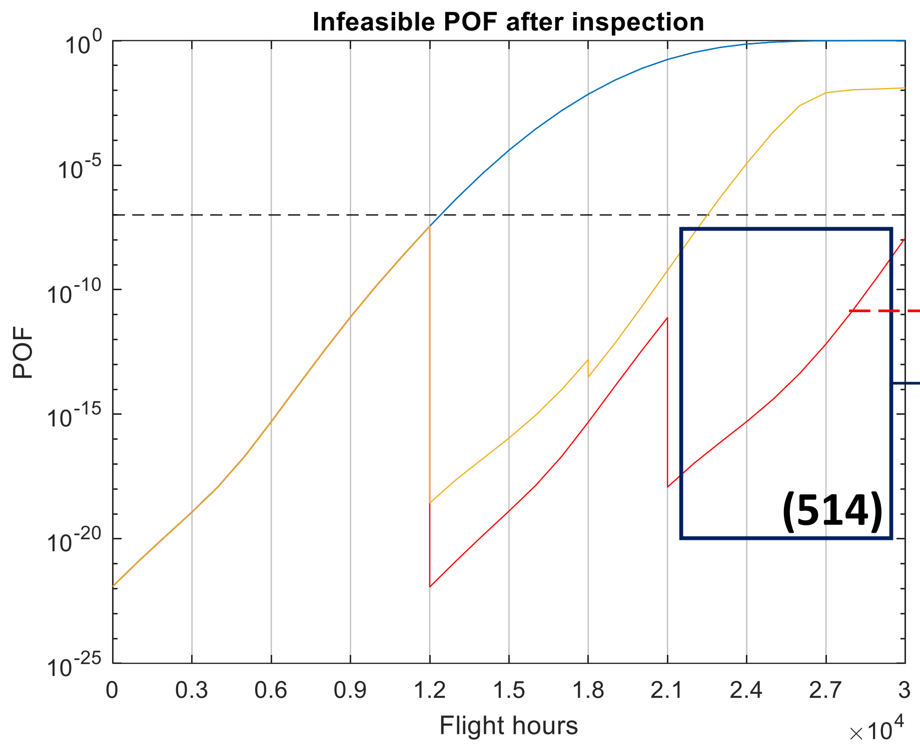
Reject and Skip Branches
Two Inspections Type



Possible inspection times [3000:3000:27000]



Feasible Branches Evaluation Two Inspection Type



POF curve for a feasible solution

Any inspection added after 21k flight hours will make the branch feasible but more expensive

Schedule times (10^3 flight hours)

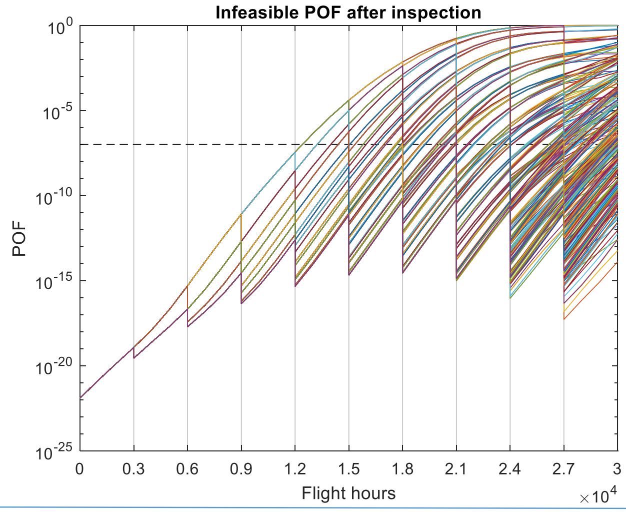
Schedule combinations	3	6	9	12	15	18	21	24	27
(1)	0	0	0	0	0	0	0	0	0
(505)	0	0	0	2	0	0	2	0	0
(506)	0	0	0	2	0	0	2	0	1
(507)	0	0	0	2	0	0	2	0	2
(508)	0	0	0	2	0	0	2	1	0
(509)	0	0	0	2	0	0	2	1	1
(510)	0	0	0	2	0	0	2	1	2
(511)	0	0	0	2	0	0	2	2	0
(512)	0	0	0	2	0	0	2	2	1
(513)	0	0	0	2	0	0	2	2	2
(514)	0	0	0	2	0	1	0	0	0

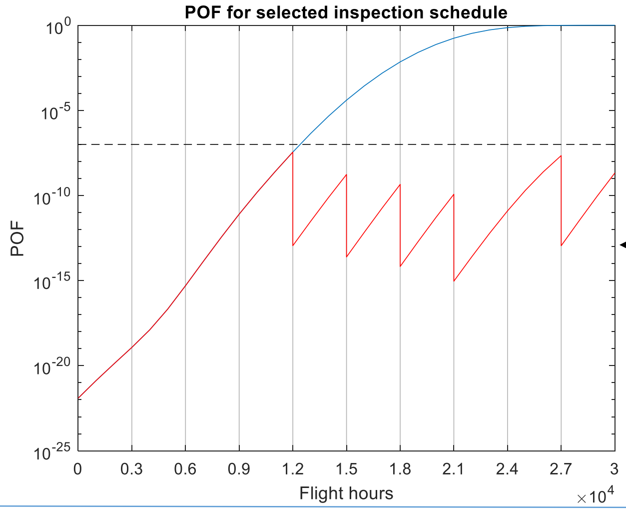
Possible inspection times [3000:3000:27000]

Method will skip POF evaluations from 505 to 514

Skipping Algorithm Validation
Single Inspection type

All combinations evaluated

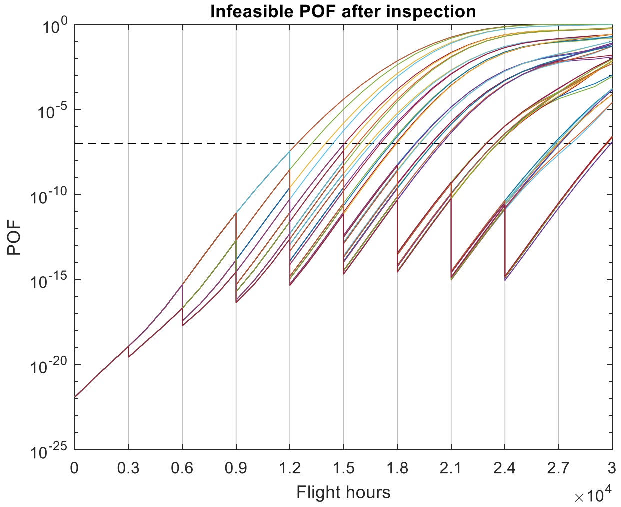


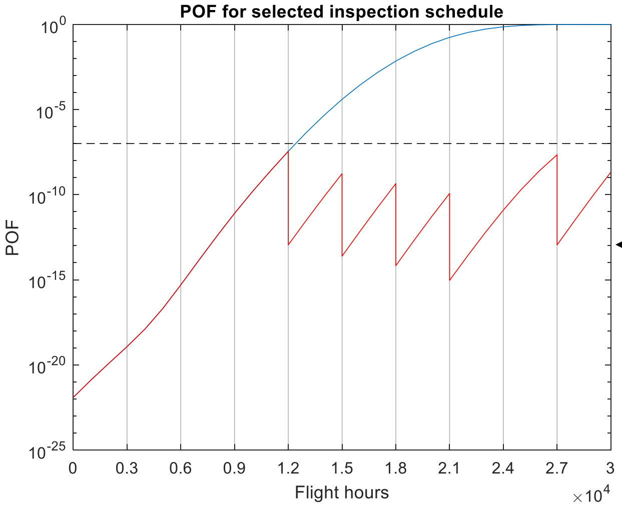


Possible inspection times [3000:3000:27000]

Same Schedule

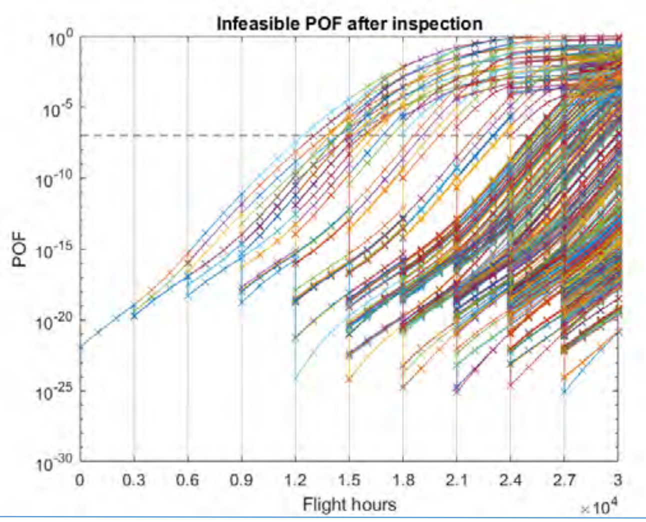
With skipping algorithm

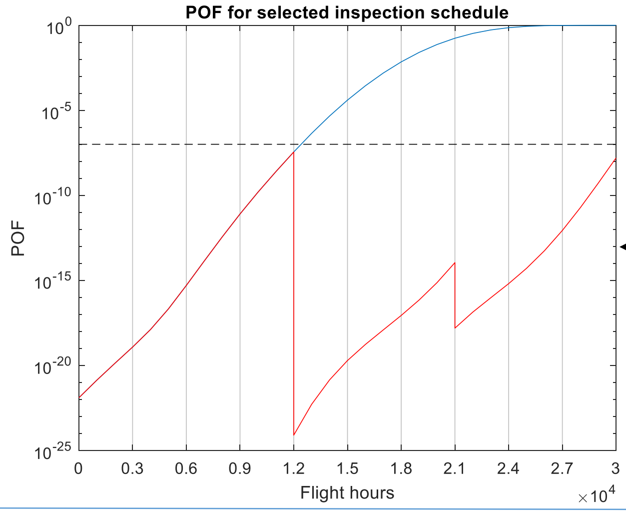




Skipping Algorithm Validation Multiple Inspection Type

All combinations evaluated

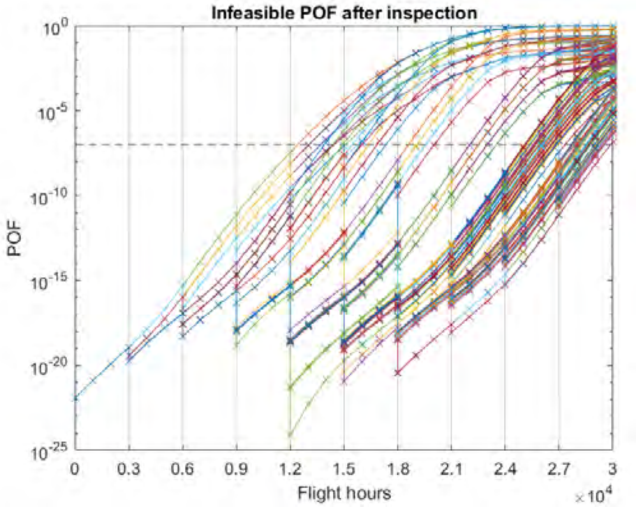


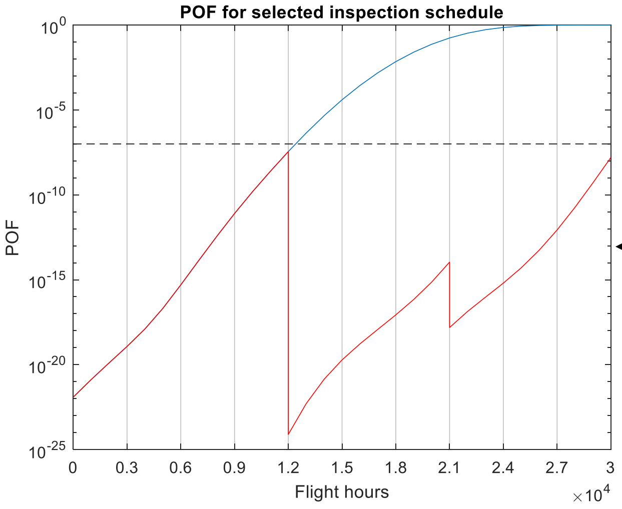


Possible inspection times [3000:3000:27000]

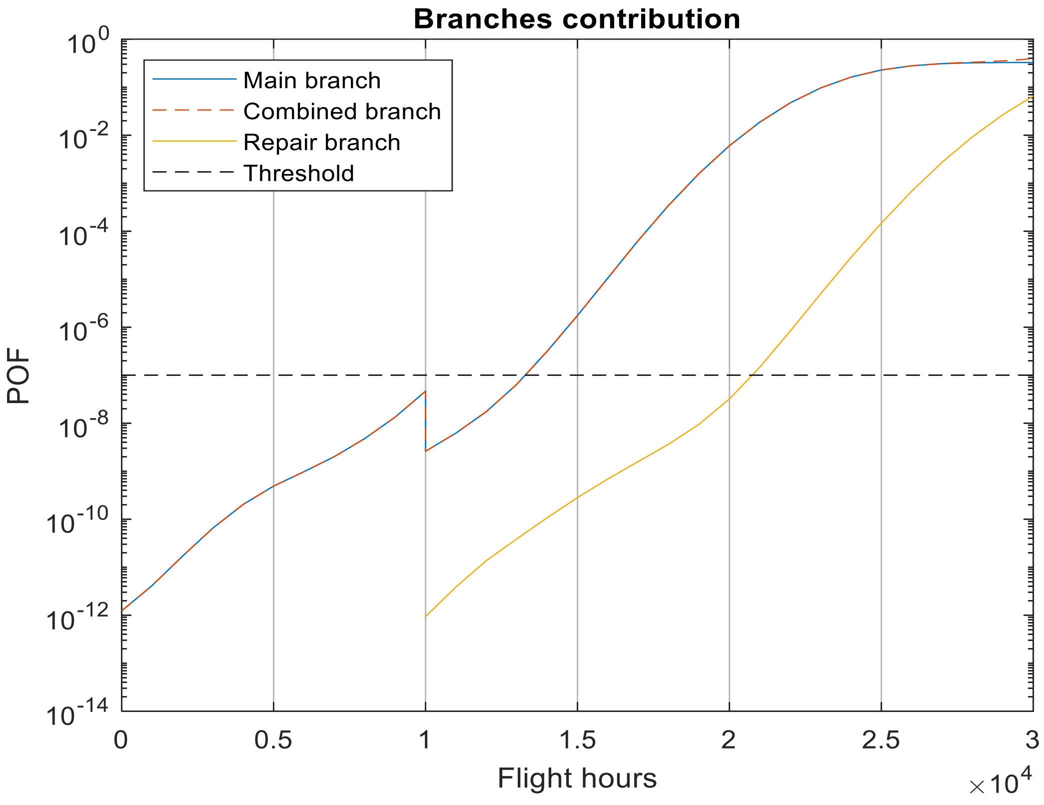
Same Schedule

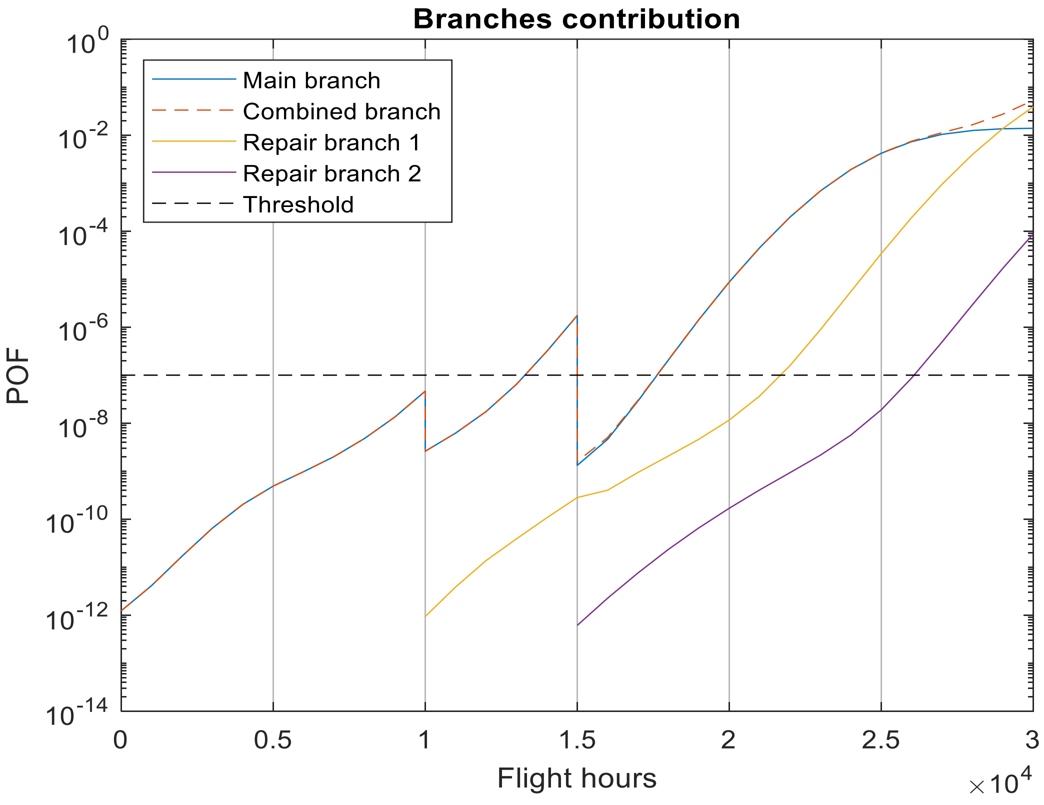
With skipping algorithm





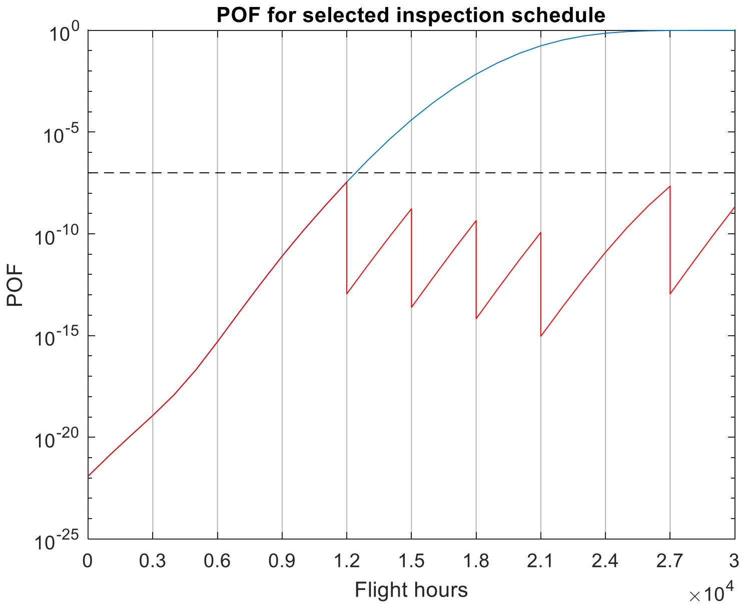
Use Main Branch Approximation

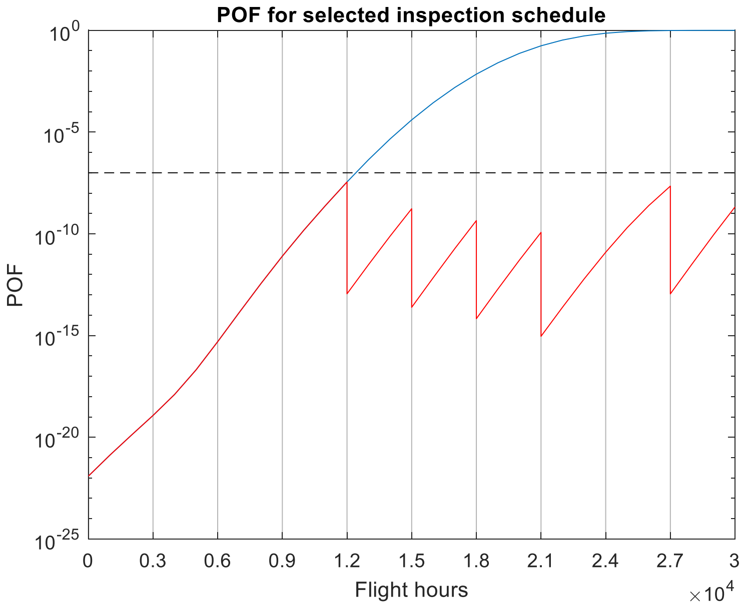


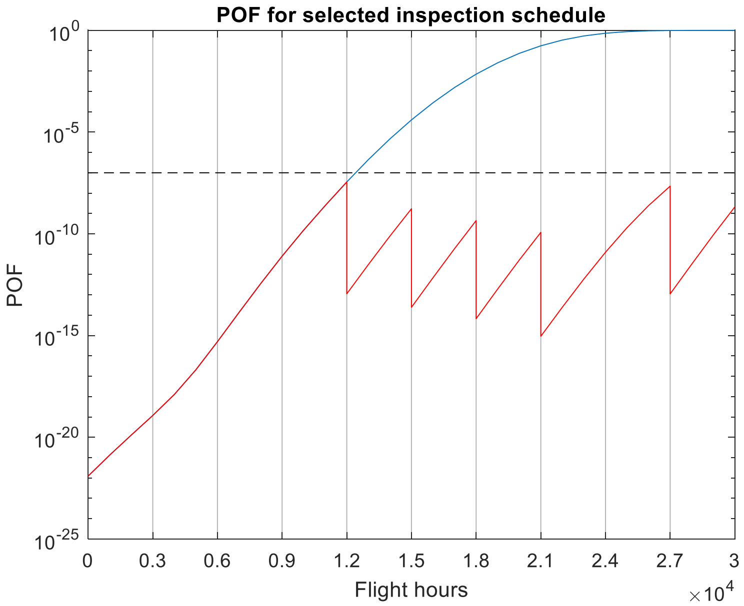


The code will only use the main branch curve information

Main Branch Approximation Validation







Possible inspection times [3000:3000:27000]
Selected inspection schedule [12000, 15000, 18000, 21000, 27000]

Without skipping algorithm

Calculation relative time: 10.5

With skipping algorithm

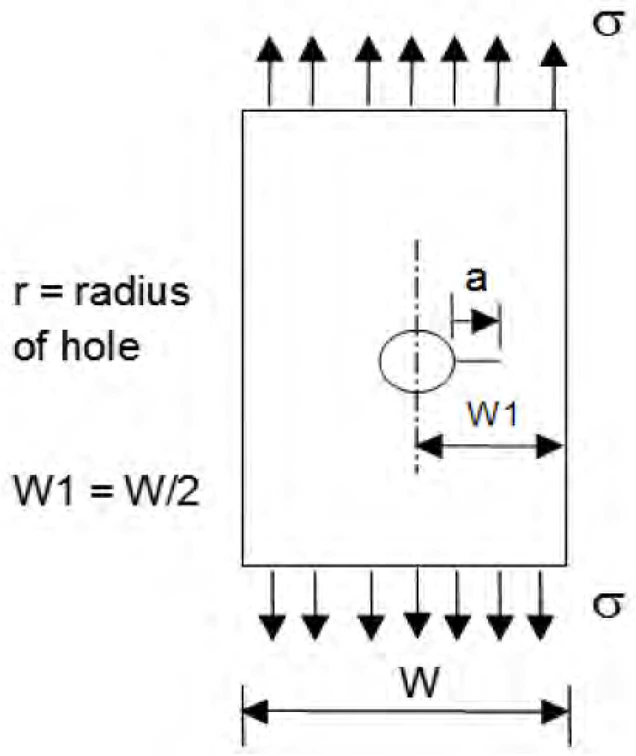
Calculation relative time: 1.7

With skipping algorithm and using main branch approximation

Calculation relative time: 1

Example

Input Data (I)



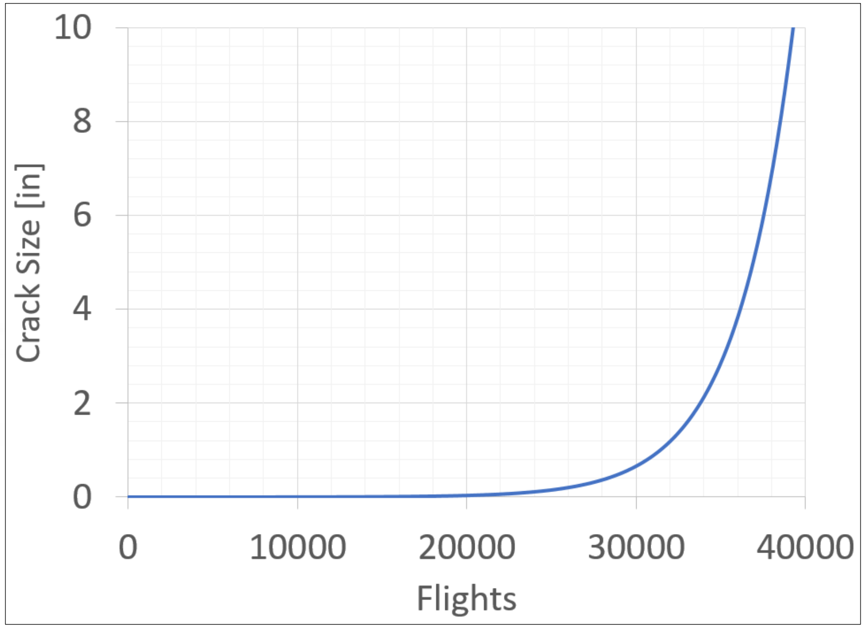
$$K = \sigma \cdot \beta \sqrt{\pi a} = \sigma \cdot \alpha$$

$$\beta = \beta_{hole} \beta_{width}$$

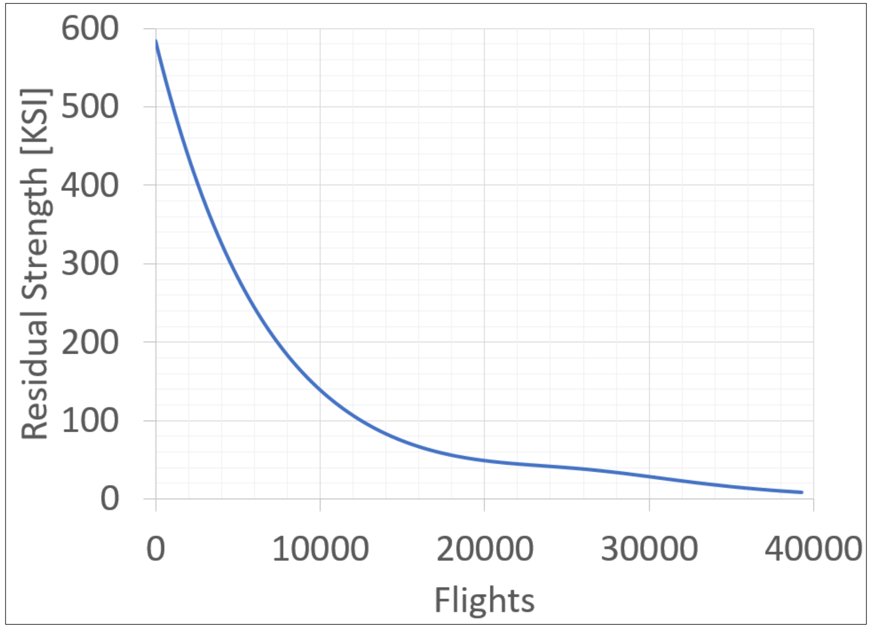
$$\beta_{hole} = 0.6762 + 0.8734 / (0.3246 + (a / R))$$

$$\beta_{width} = \sqrt{\sec\left(\frac{\pi(R+a)}{W}\right)}$$

Crack Growth



Residual Strength

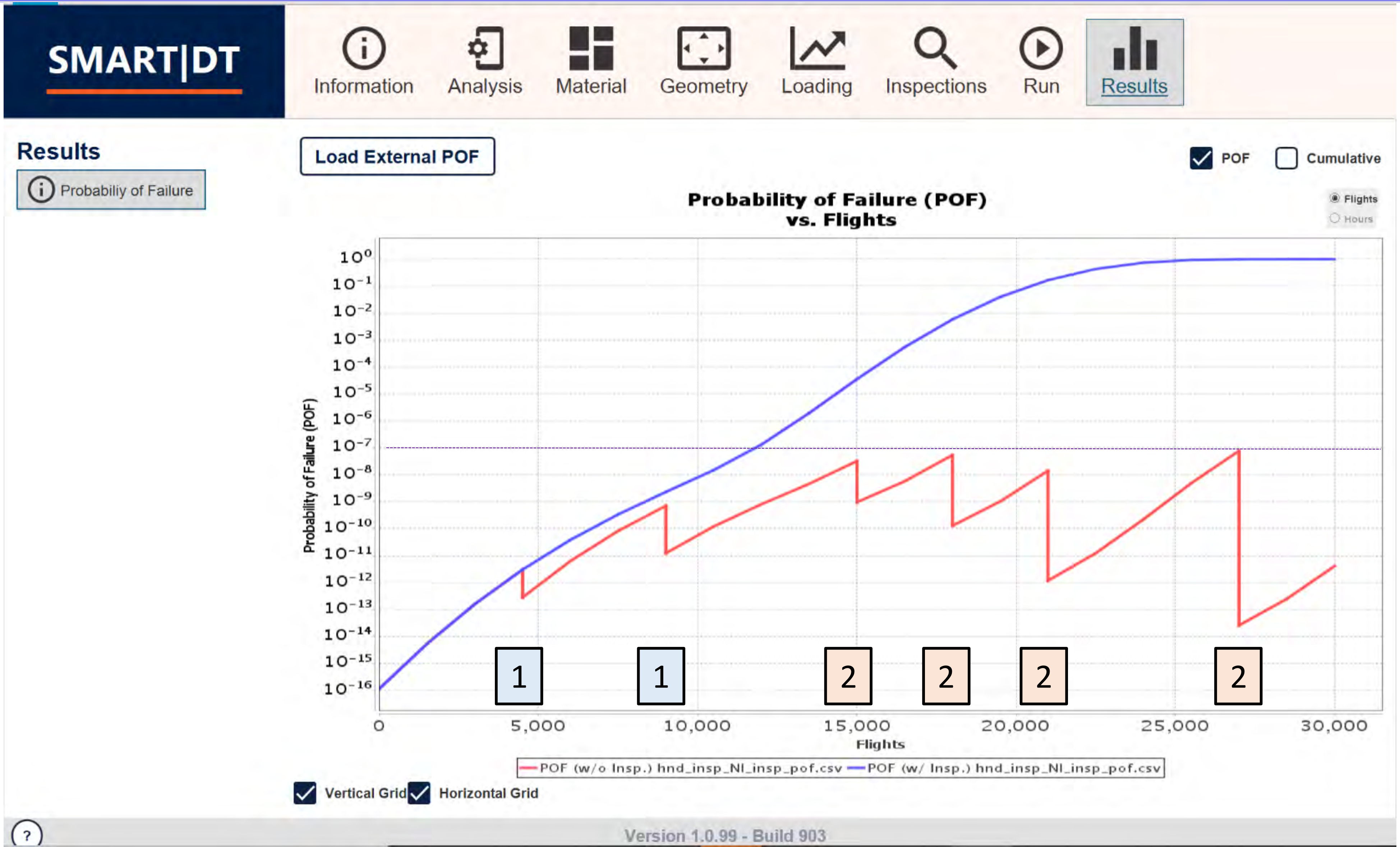


Input Data (II)

Variable	Dist. Type	mean	St. Dev.	Notes
Initial Crack Size	Lognormal	0.00248 in	0.00129	Reamed Fastener Hole
Repair Crack Size	Lognormal	0.00248 in	0.00129	Assuming Repair is Replacement of Part
Fracture Toughness	Normal	26.0 ksi	2.0	7050-T651 Plate
EVD	Gumbel	14.5 ksi	0.8	

Inspections	Inspection Type	Material	Crack Type	Dist. Type	Mean [in]	St. Dev. [in]	Source	Cost
POD 1	Automated bolt hole eddy current	Aluminum	T	Lognormal	0.0179	0.0108	Aeronautical Applications of Non-destructive	5x
POD 2	Eddy current sliding probe	Aluminum	Overall	Lognormal	0.0788	0.0302	NDE Capabilities Book	1x

Results


SMART|DT
Information
Analysis
Material
Geometry
Loading
Inspections
Run
Results
Results
Probabiliy of Failure
Load External POF
POF
Cumulative
Probability of Failure (POF)
vs. Flights
Flights
Hours
Probability of Failure (POF)
Flights
1
1
2
2
2
2
POF (w/o Insp.) hnd_insp_NI_insp_pof.csv
POF (w/ Insp.) hnd_insp_NI_insp_pof.csv
Vertical Grid
Horizontal Grid
Version 1.0.99 - Build 903

Bonus

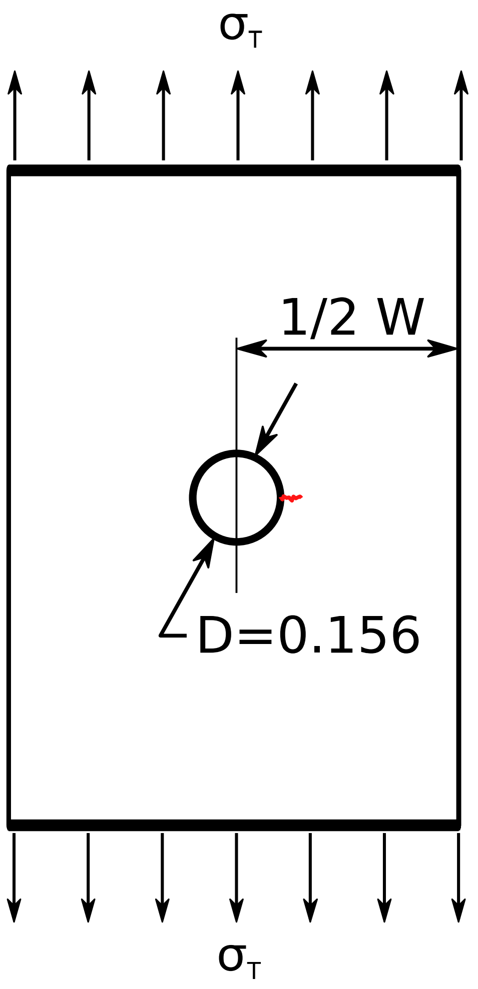
Cross-Entropy Based Adaptive Importance Sampling for Probabilistic Damage Tolerance Analysis

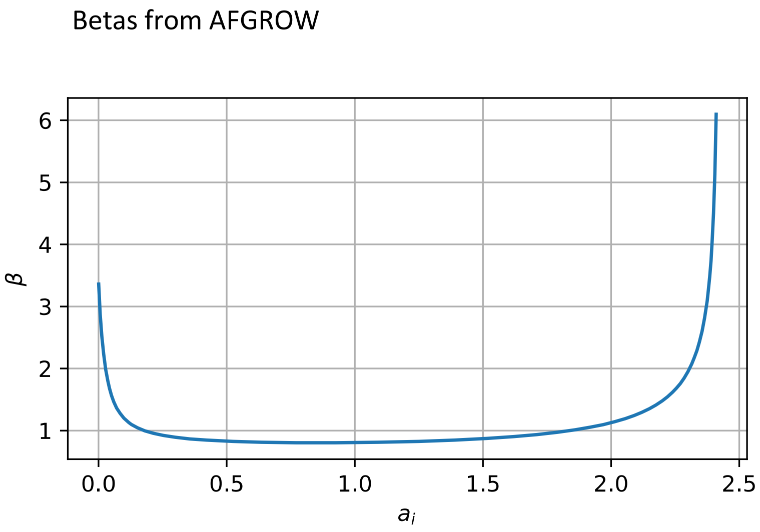
AA&S 2020

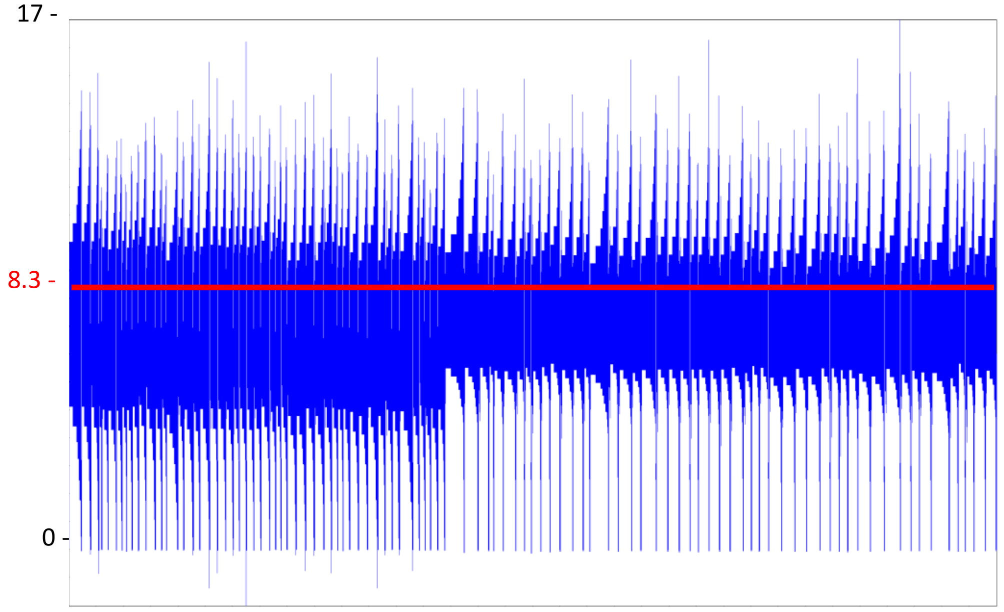
Adaptive Cross Entropy Method

- Goal: minimize the Kullback-Leibler Cross Entropy
  - $\mathcal{D}(g,h) = \mathbb{E}_{f^*}\left[-\ln\left(\frac{f^*(X)}{h(X;\theta)}\right)\right] = \int \ln\big(f^*(X)\big)\, f^*(X)\,\mathrm{d}x - \int \ln\big(h(X;\theta)\big)\, f^*(X)\,\mathrm{d}x$
    - $f^*(X)$ is the estimated optimal sampling density
    - $h(X;\theta)$ is a PDF with parameter vector θ
    - First integral involves only $f^*(X)$, so evaluates to a constant
    - Maximizing the second integral yields the optimal parameter vector θ^*
  - Optimization problem
    - $-\ln(\cdot)$ is convex, so ideal for optimization
    - $\theta^* = \underset{\theta}{\operatorname{argmax}}\left[\frac{1}{N}\sum_{i=1}^{N} f^*(x_i)\ln\big(h(x_i;\theta)\big)\right] \implies \text{solve } \frac{1}{N}\sum_{i}^{N} f^*(x_I)\,\nabla_\theta \ln\big(h(x_i;\theta)\big) = 0$
  - Closed form solutions for many distributions, especially Natural Exponential Family
    - For multivariate normal distribution: $\hat{\theta}^* = \{\hat{\mu},\hat{\Sigma}\}$, $\hat{\mu} = \frac{\sum_{i=1}^{N} f^*(x_i)\,x_i}{\sum_{i=1}^{N} f^*(x_i)}$, $\hat{\Sigma} = \frac{\sum_{i}^{N} f^*(x_i)(x_i-\hat{\mu})^2}{\sum_{i=1}^{N} f^*(x_i)}$
  - Converges after a finite number of iterations

Example Problem

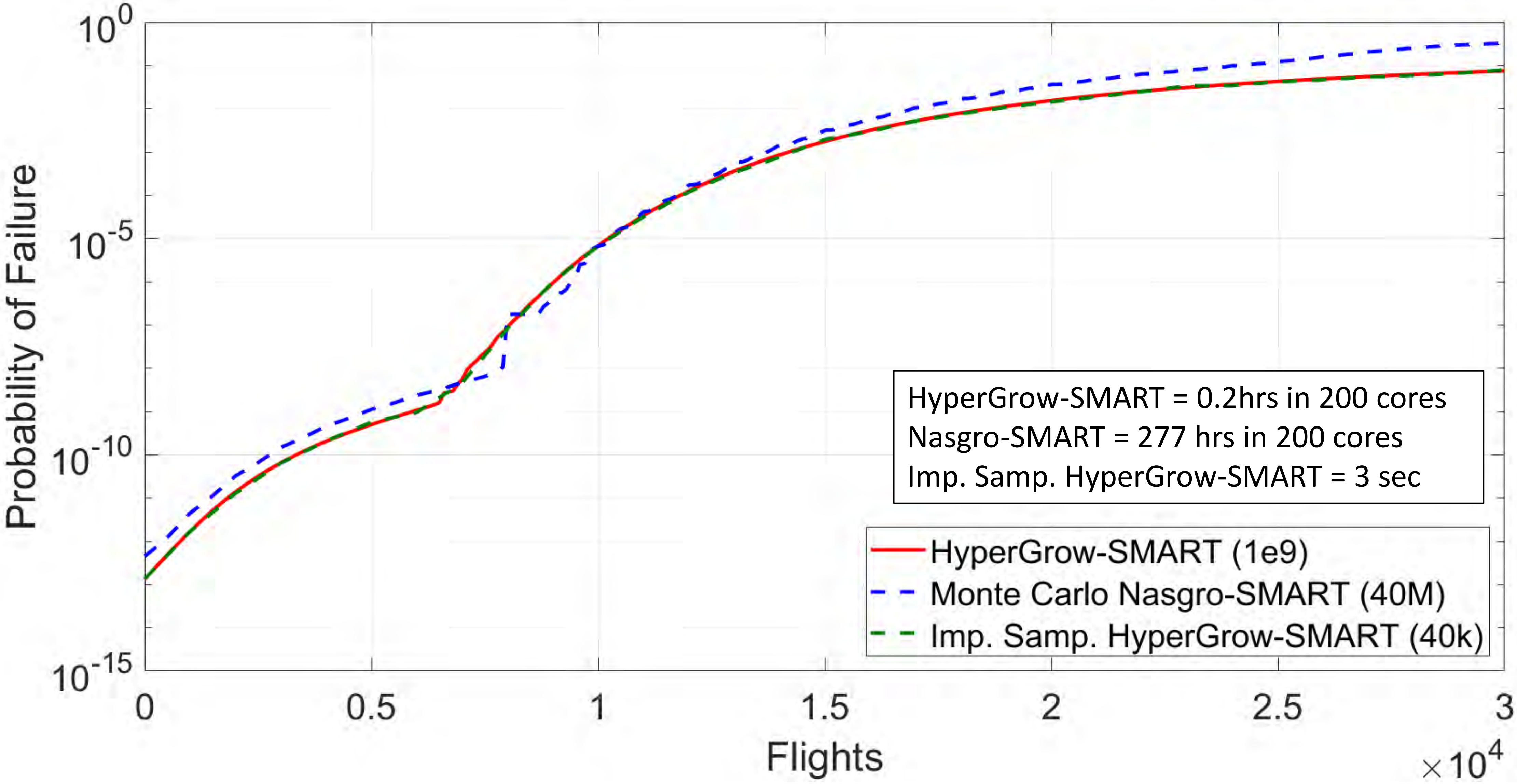






Geometry	Deterministic(5 wide x 0.125 thick)	in
Initial Crack Size	Weibull(α=0.45, β=4.17e-5)	in
Fracture Toughness	Normal(μ=35.0, σ=3.5)	ksi √in
log(Paris C)	Normal(μ=-9.0, σ=0.08)	
Paris n	3.8	
Maximum Load	Frechet(μ=13.4, σ=1.29, ξ=0.07)	ksi

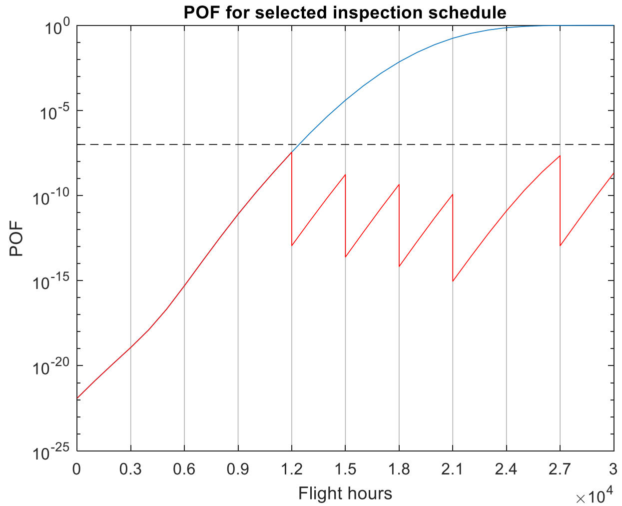
Results

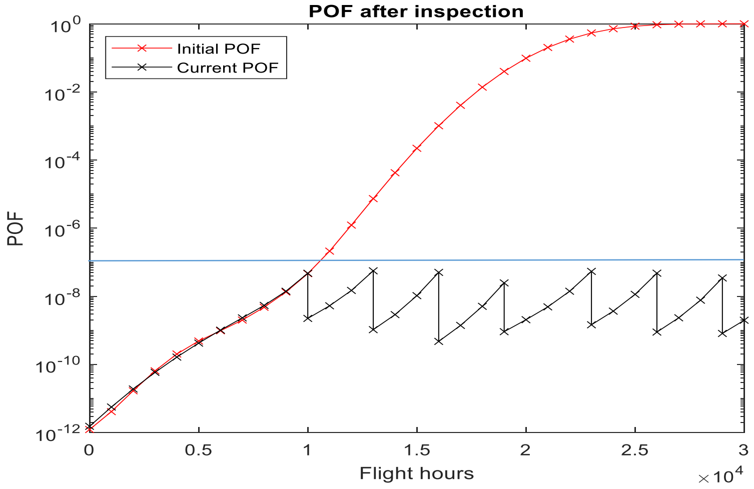


Probability of Failure
10^{0}
10^{-5}
10^{-10}
10^{-15}
0
0.5
1
1.5
2
2.5
3
Flights
$\times 10^{4}$
HyperGrow-SMART = 0.2hrs in 200 cores
Nasgro-SMART = 277 hrs in 200 cores
Imp. Samp. HyperGrow-SMART = 3 sec
HyperGrow-SMART (1e9)
Monte Carlo Nasgro-SMART (40M)
Imp. Samp. HyperGrow-SMART (40k)

Future Work

- Finish the Shortest Path Method (SPM) implementation in SMART.
- Implement OpenMP and MPI to the SPM.
- Continue looking for alternatives to speed up the calculations (Still very slow).

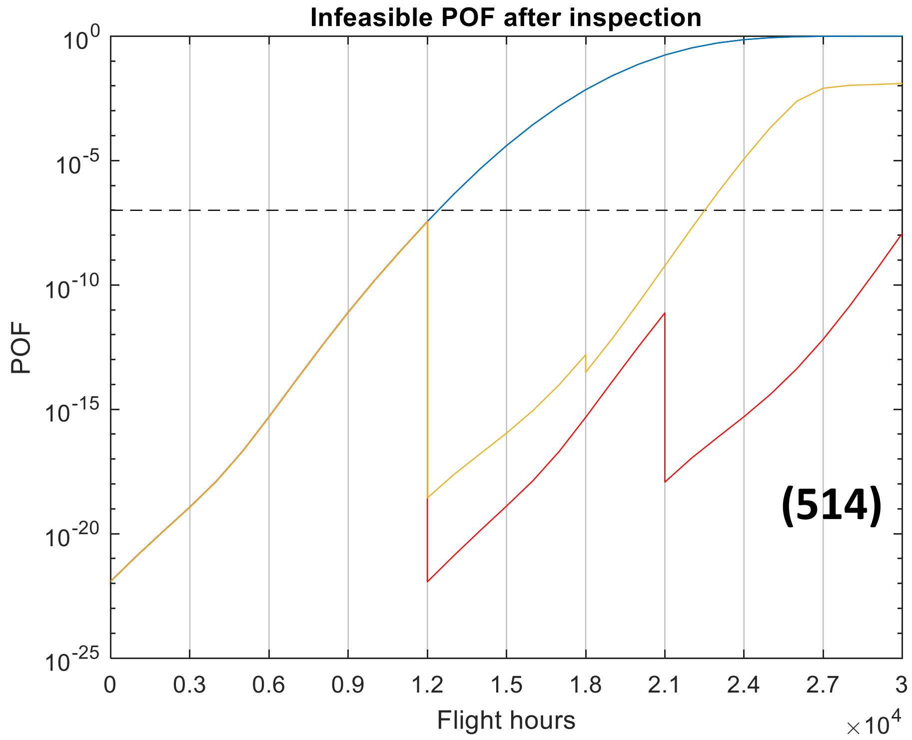




Acknowledgements

Probabilistic Fatigue Management Program for General Aviation, Federal Aviation Administration, Grant 12-G-012

Skip Evaluation for Feasible Branches Two Inspection Type



Possible inspection times [3000:3000:27000]