Risk Based Optimized Inspections for Aircraft Fleets

Juan D. Ocampo

St. Mary's University

Harry Millwater and Daniela Posso-Hoyos

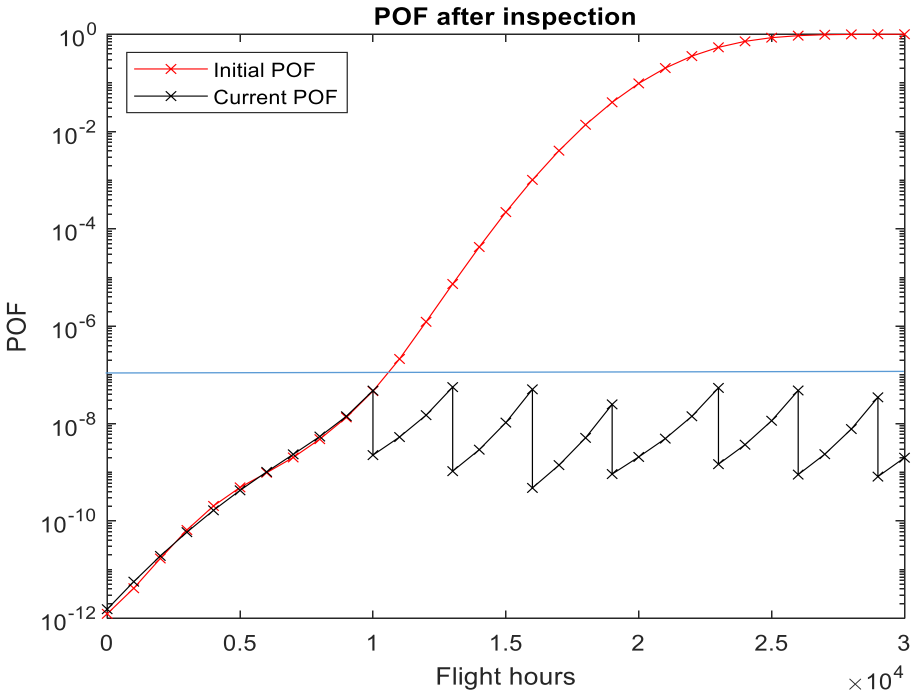
University of Texas at San Antonio

AA&S 2020 VIRTUAL

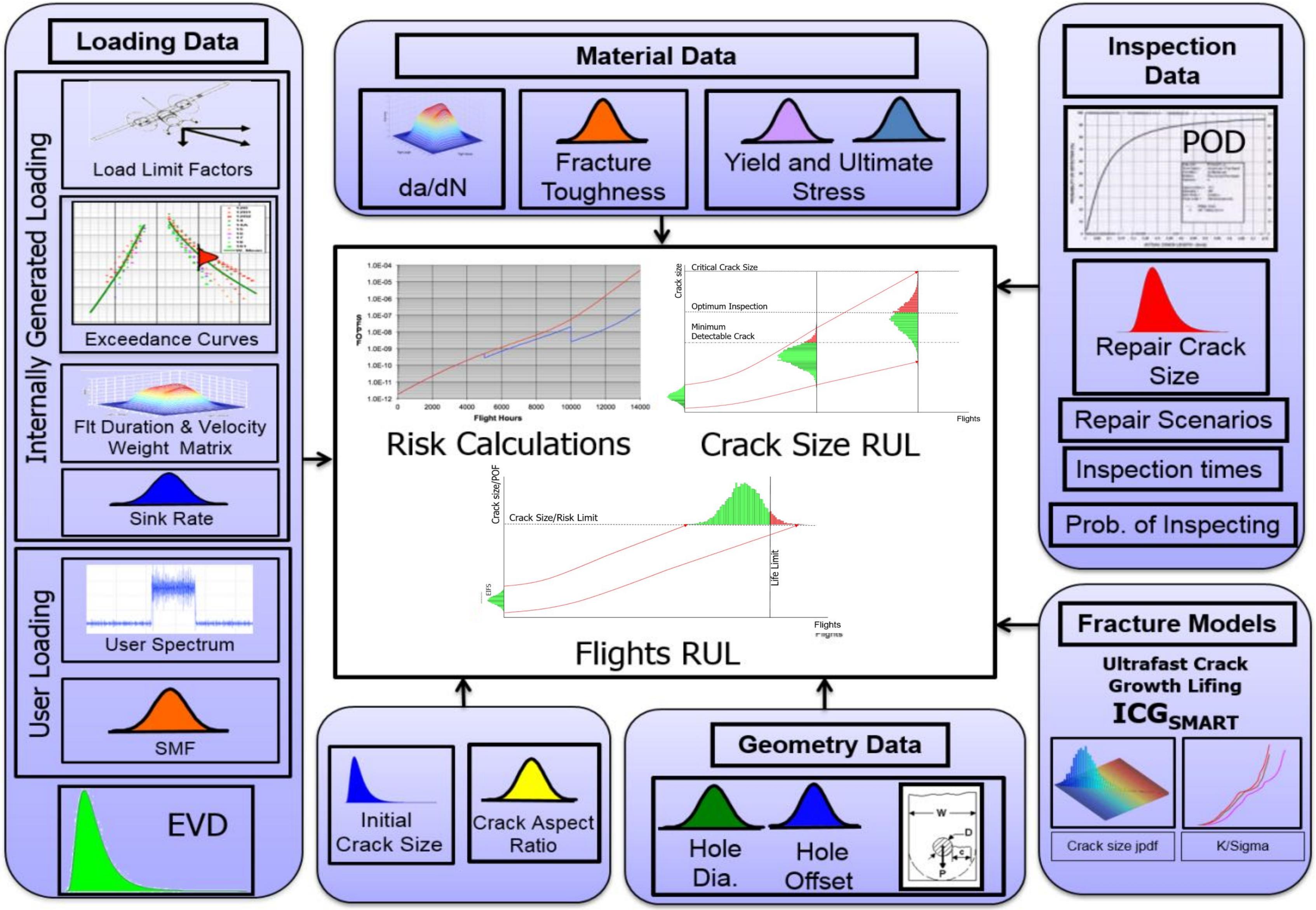
August 25-28, 2020

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

Crack Size RUL

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

Motivation

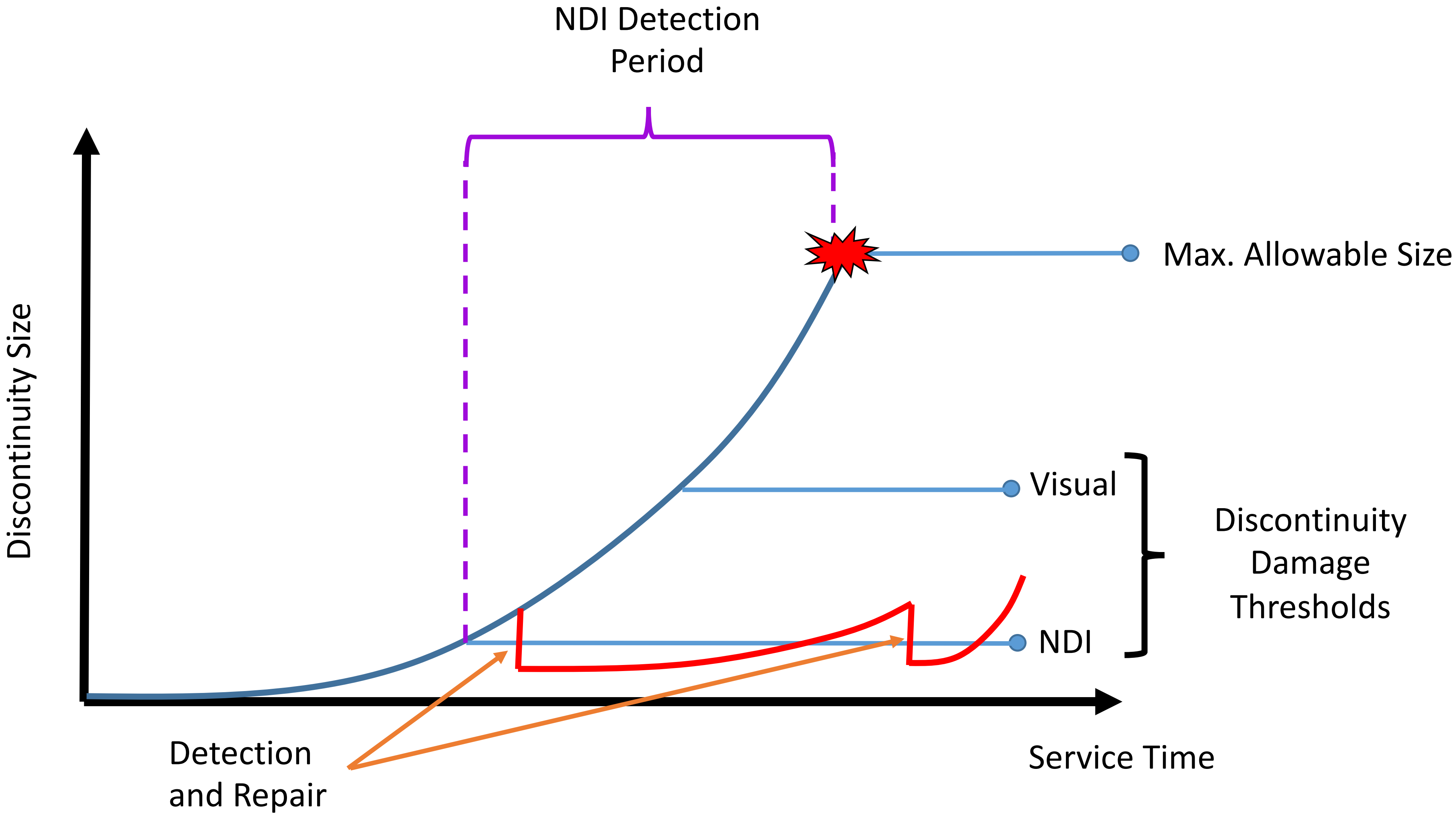
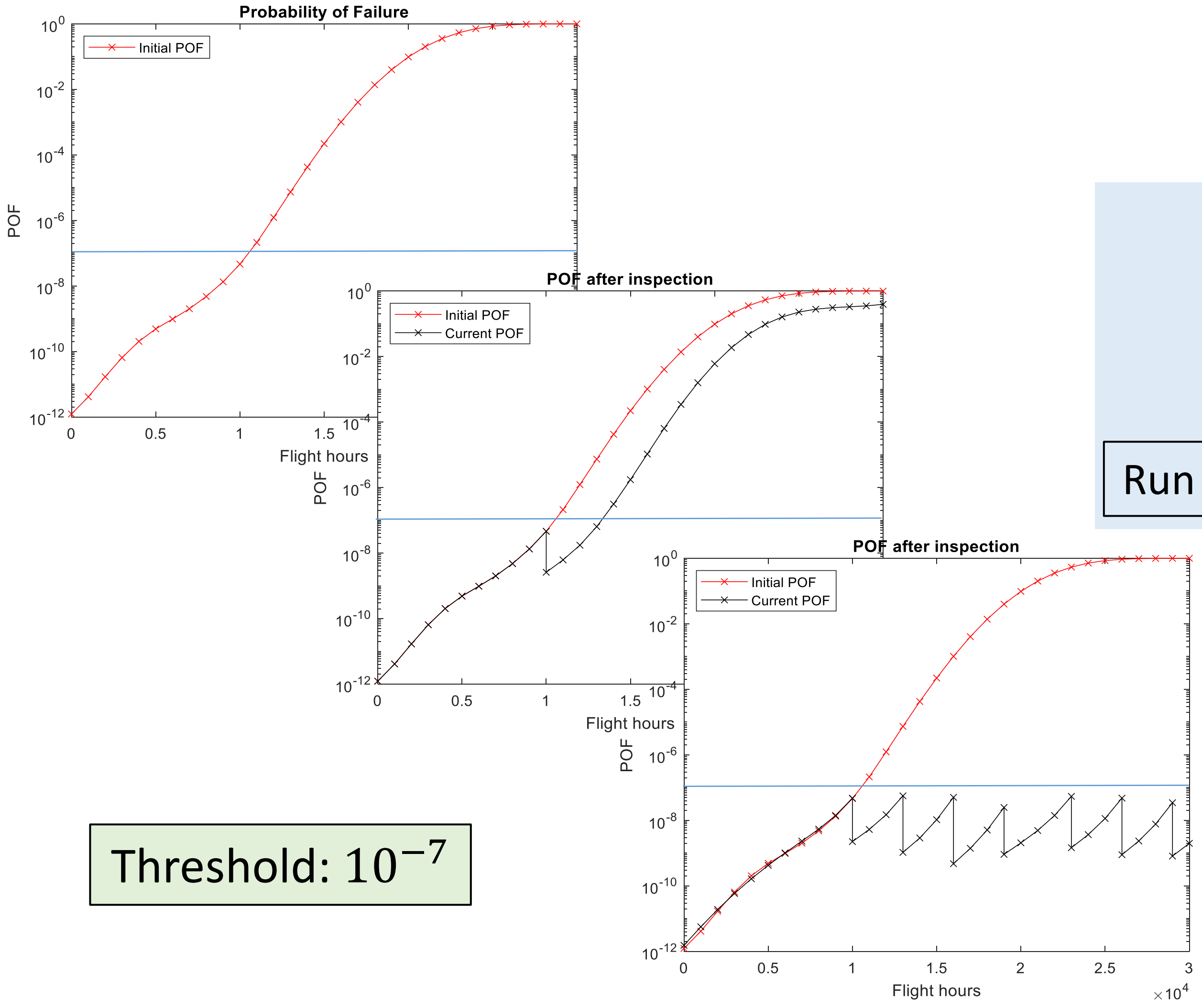

NDI Detection
Period
Max. Allowable Size
Discontinuity Size
Visual
Discontinuity
Damage
Thresholds
NDI
Detection
and Repair
Service Time

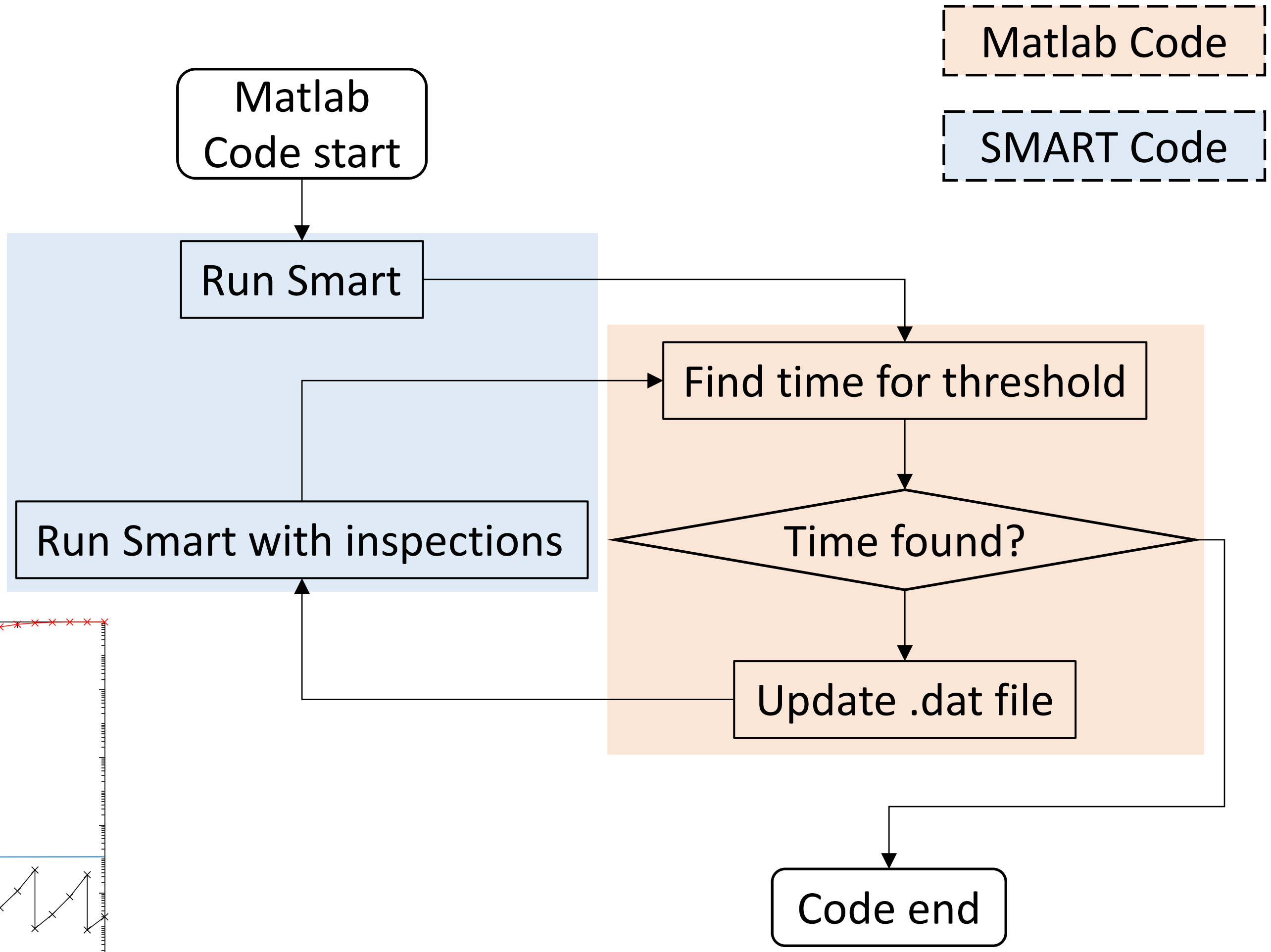
Table of capabilities

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

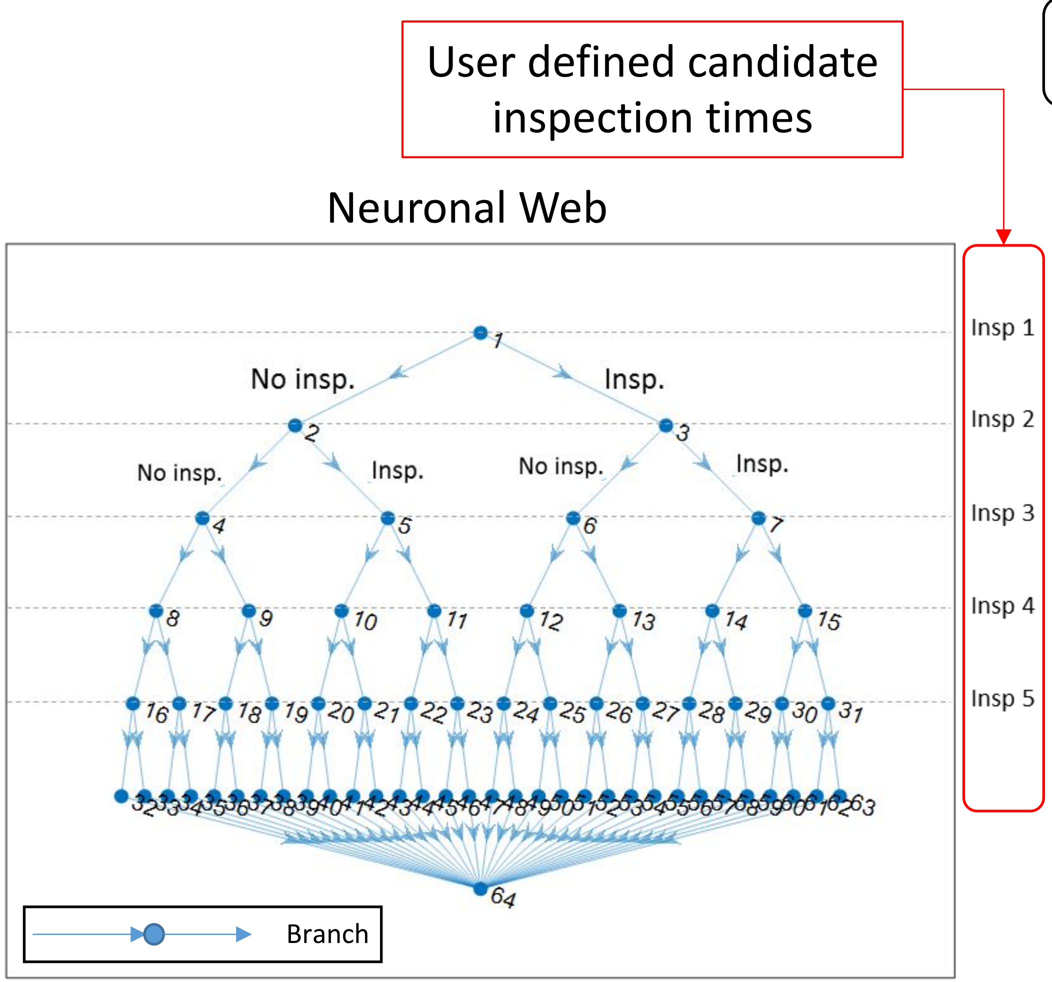
Constant risk threshold



Threshold: 10^{-7}

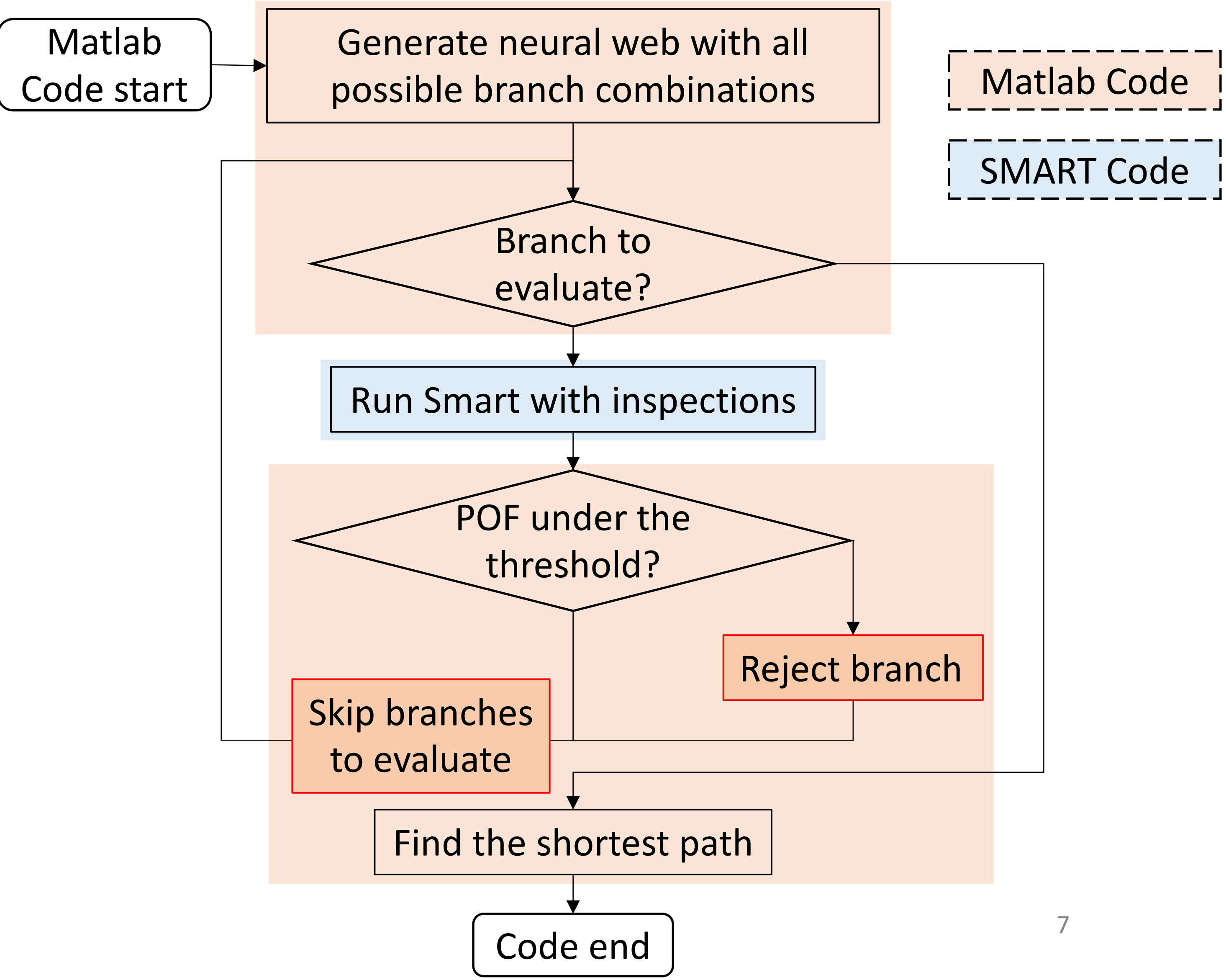


Shortest Path Method



User defined candidate inspection times

Neuronal Web



Matlab Code start

Generate neural web with all possible branch combinations

Branch to evaluate?

Run Smart with inspections

POF under the threshold?

Reject branch

Skip branches to evaluate

Find the shortest path

Code end

Matlab Code

SMART Code

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

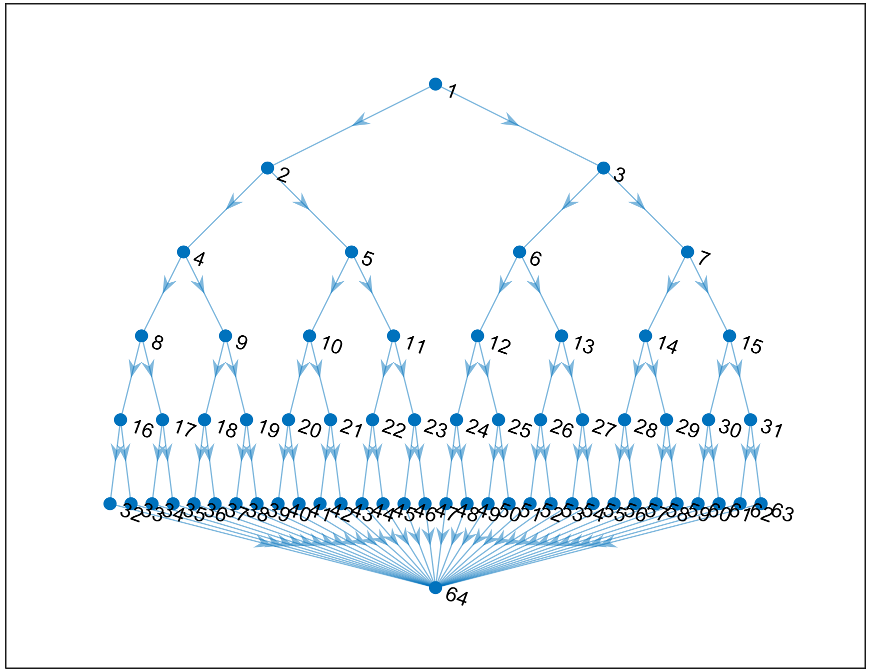
If $\qquad F_k > \ threshold$

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

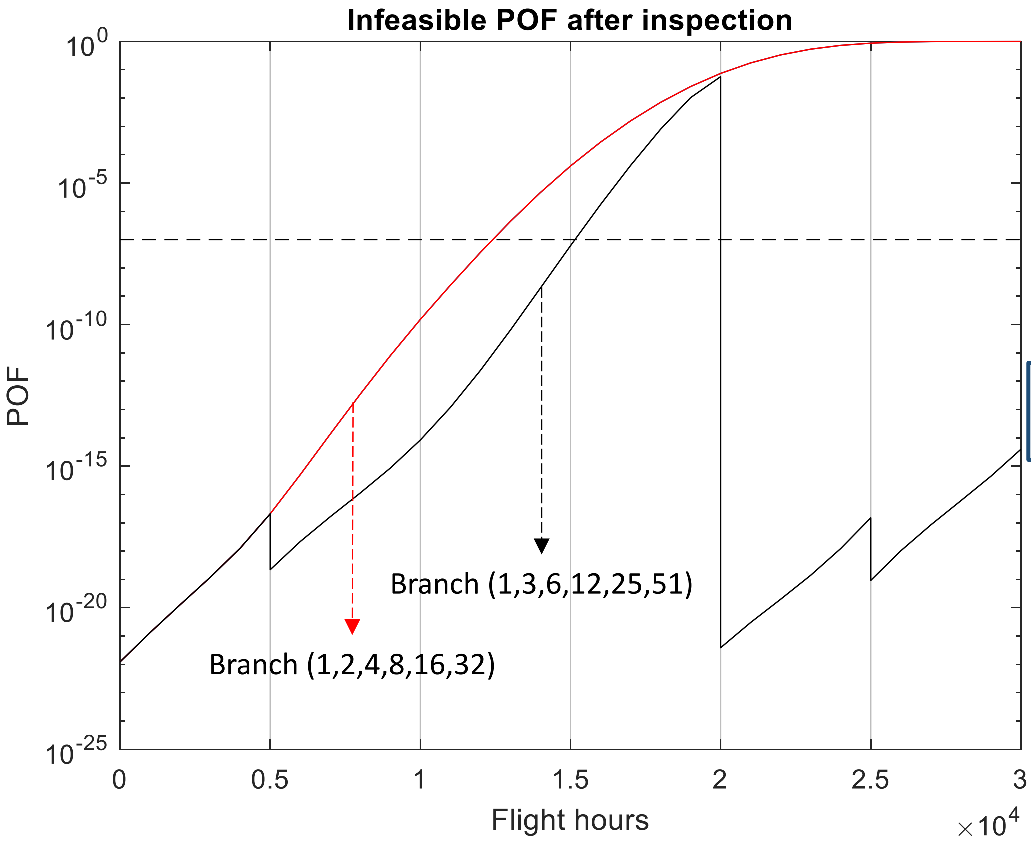
$\qquad C_{ij} = M$

$\qquad X_{ij} \le F_i, \forall\, j$

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

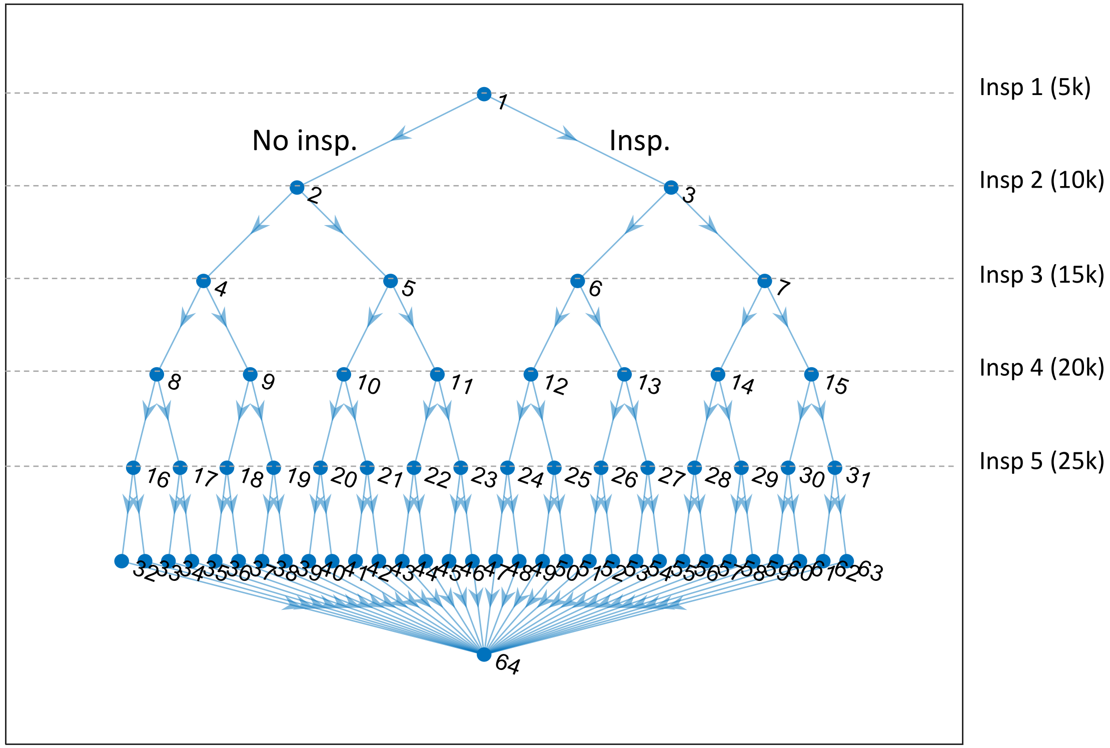


Shortest Path - Single Inspection



Visualization example

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

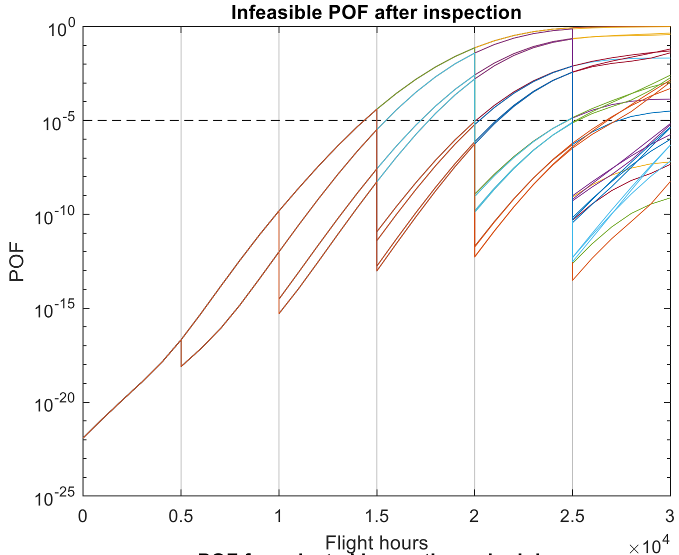


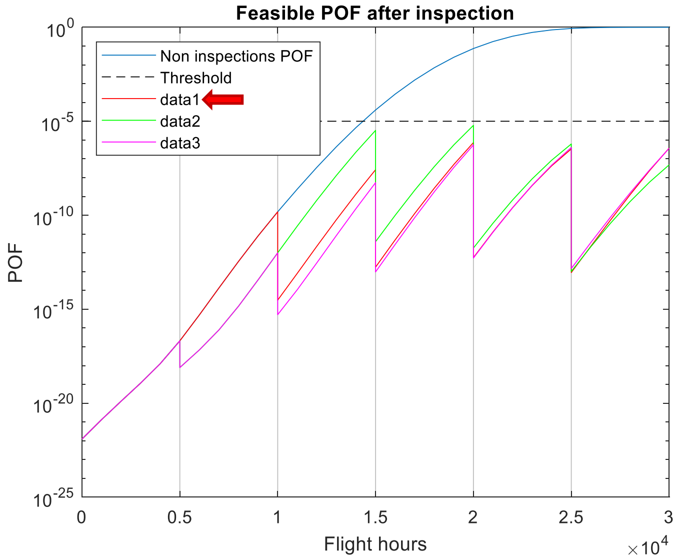
POFs for each branch / inspection schedule

32 SMART Runs

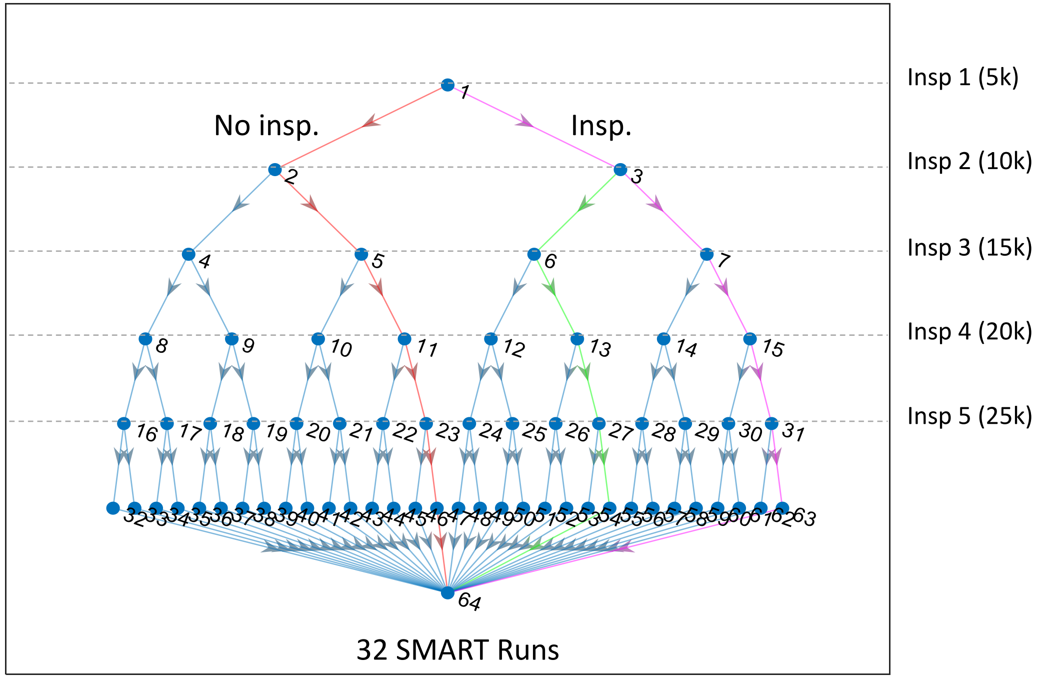
Threshold: 10^{-7}

Shortest Path - Single Inspection

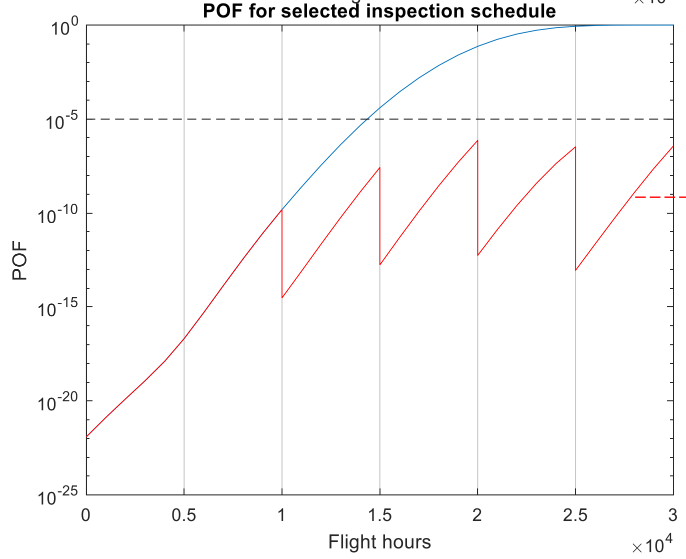

Infeasible POF after inspection


Feasible POF after inspection

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



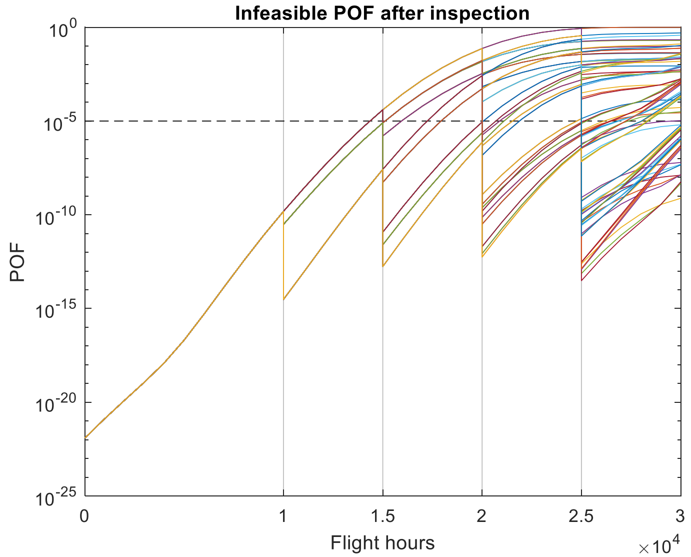
Possible inspection times [5000:5000:25000]

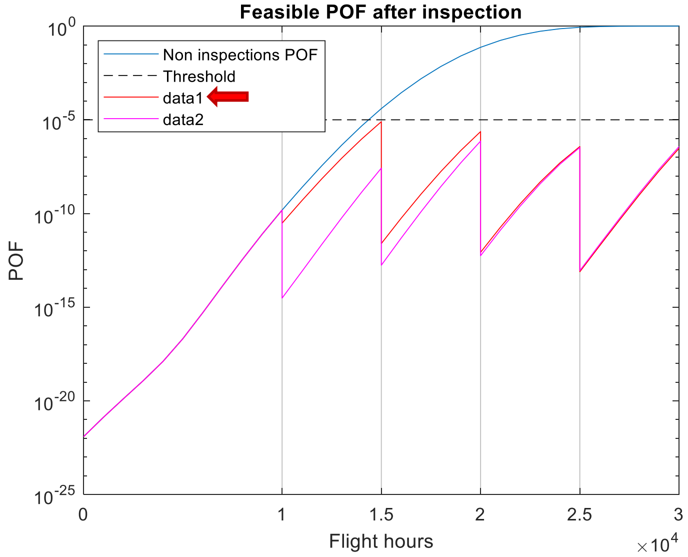

POF for selected inspection schedule

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

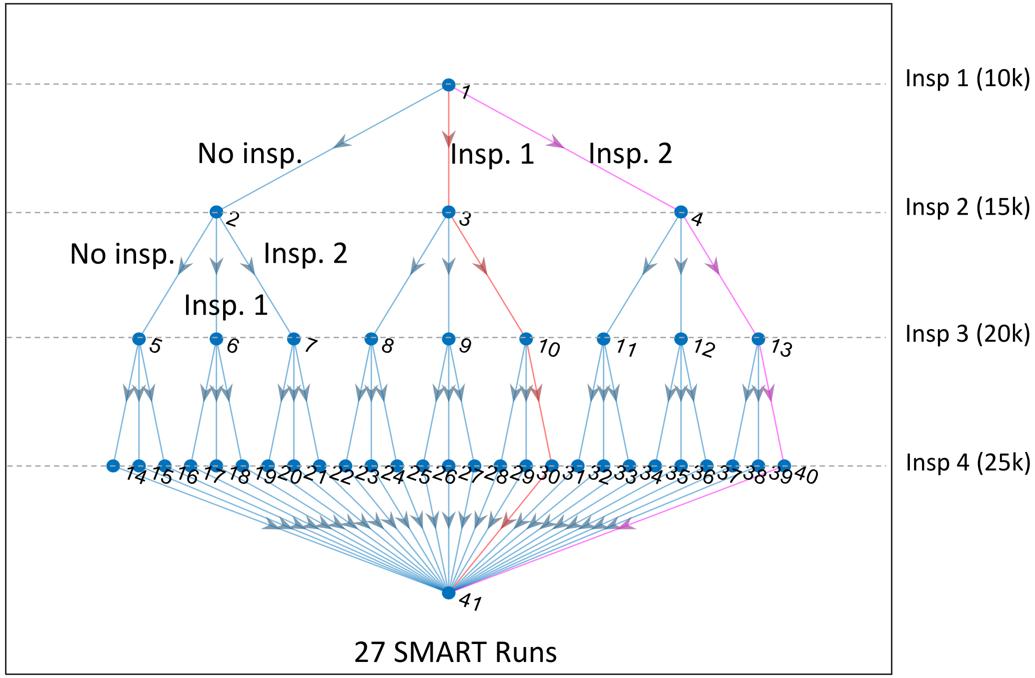
Selected schedule [10000,15000,20000,25000]

Shortest Path - Single Inspection

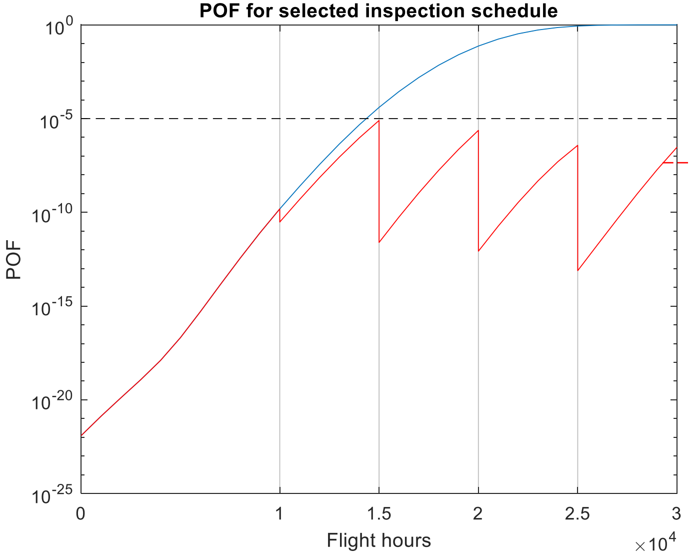




User defined inspections at: 10k, 15k, 20k, and 25k



27 SMART Runs



Selected branch
(1,3,10,31)

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

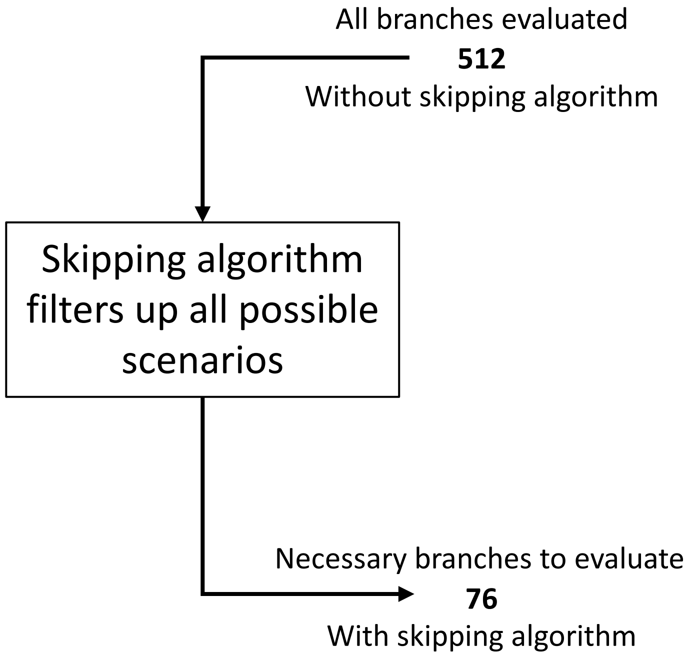
Possible inspection times [10000,15000,20000,25000]

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

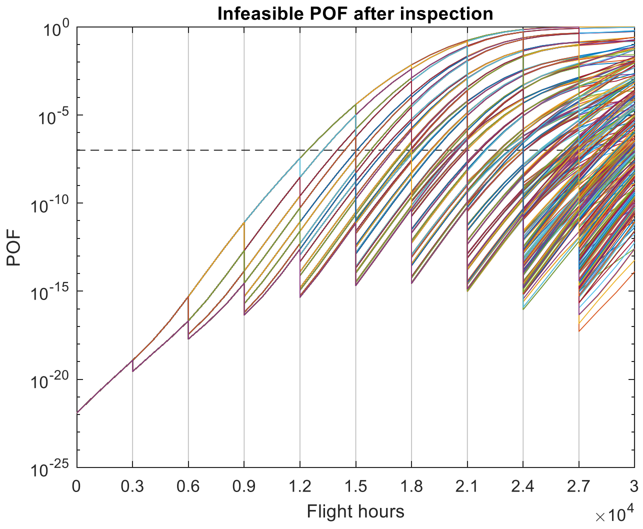
Skipping algorithm filters up all possible scenarios

Necessary branches to evaluate

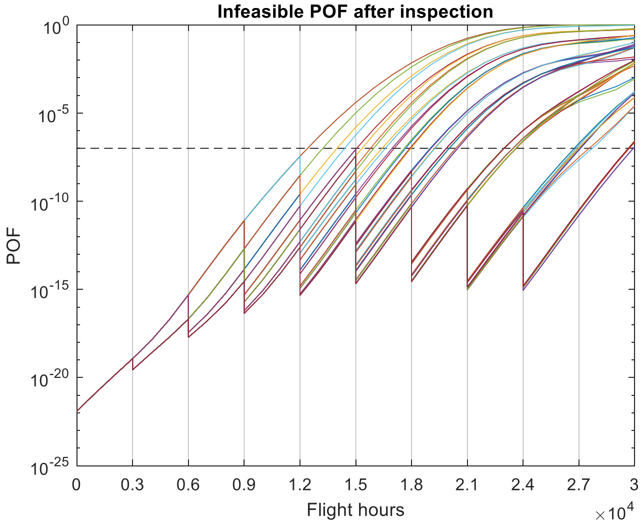
76

With skipping algorithm

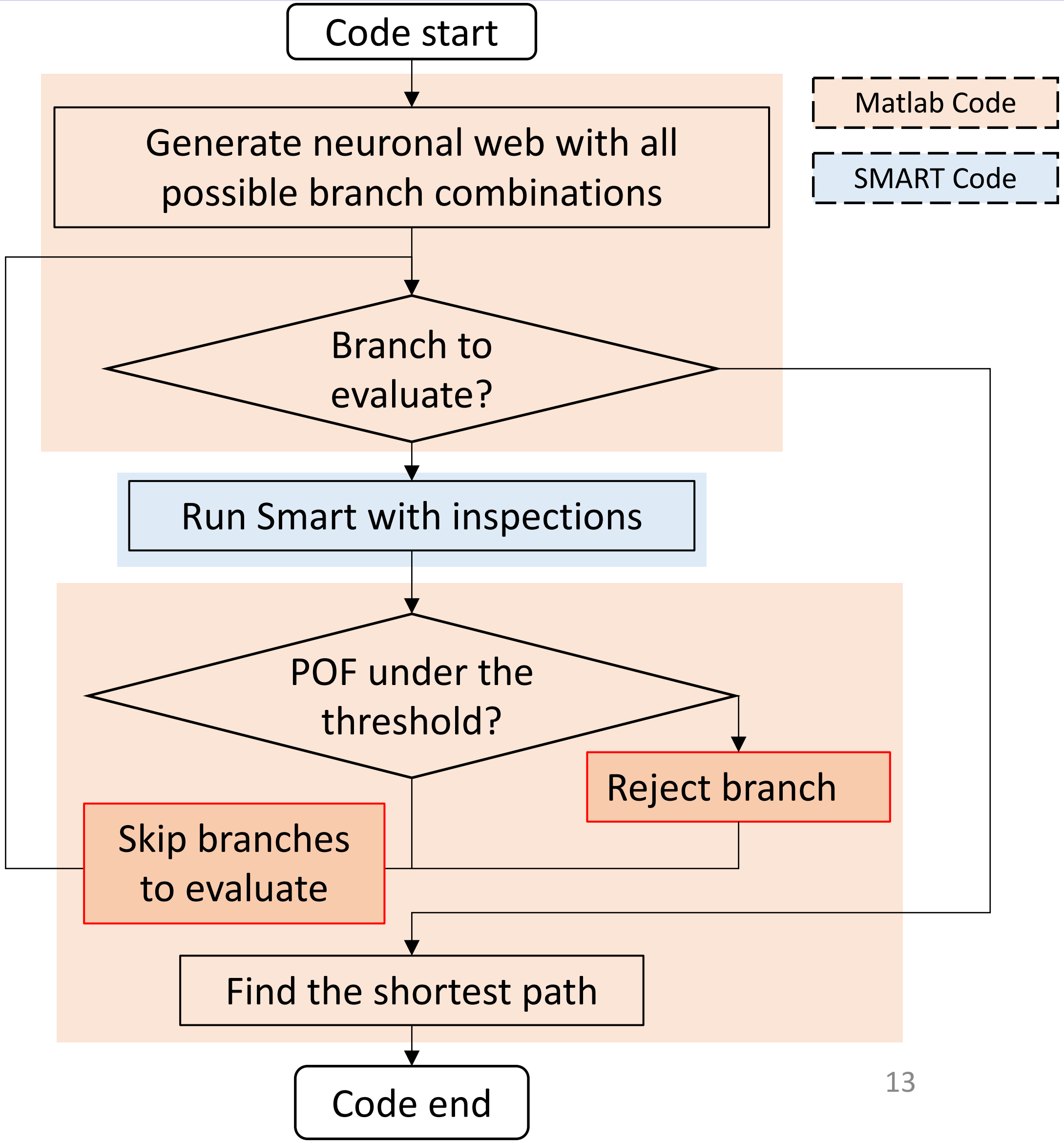
Threshold: 10^{-7}



Infeasible POF after inspection



Infeasible POF after inspection



Code start

Matlab Code

SMART Code

Generate neuronal web with all possible branch combinations

Branch to evaluate?

Run Smart with inspections

POF under the threshold?

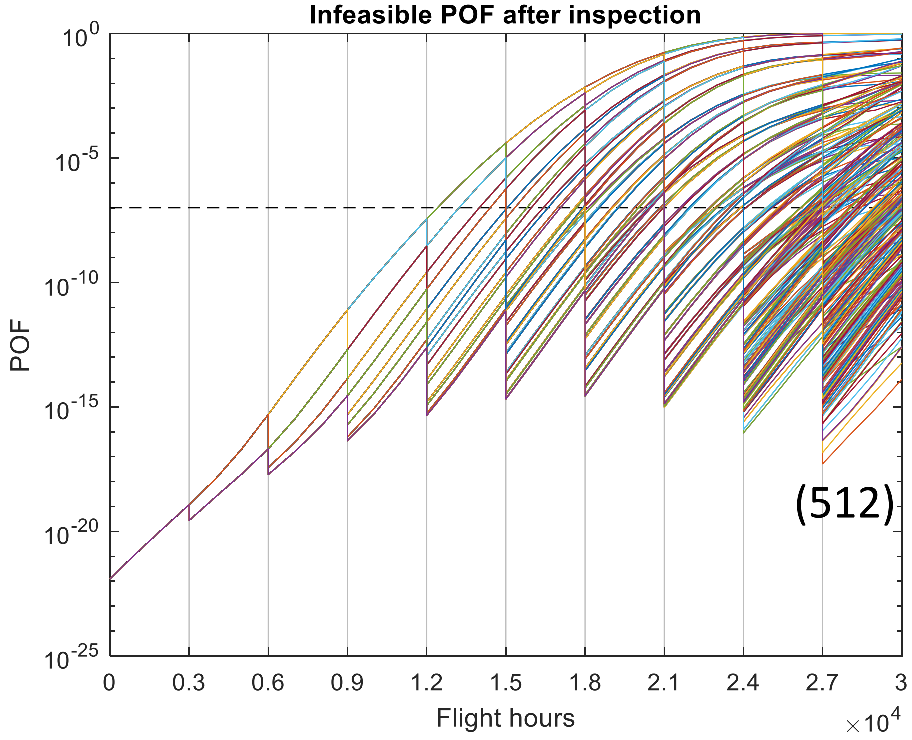
Reject branch

Skip branches to evaluate

Find the shortest path

Code end

Inspection Combination Matrix
One Inspection Type



Possible inspection times [3000:3000:27000]

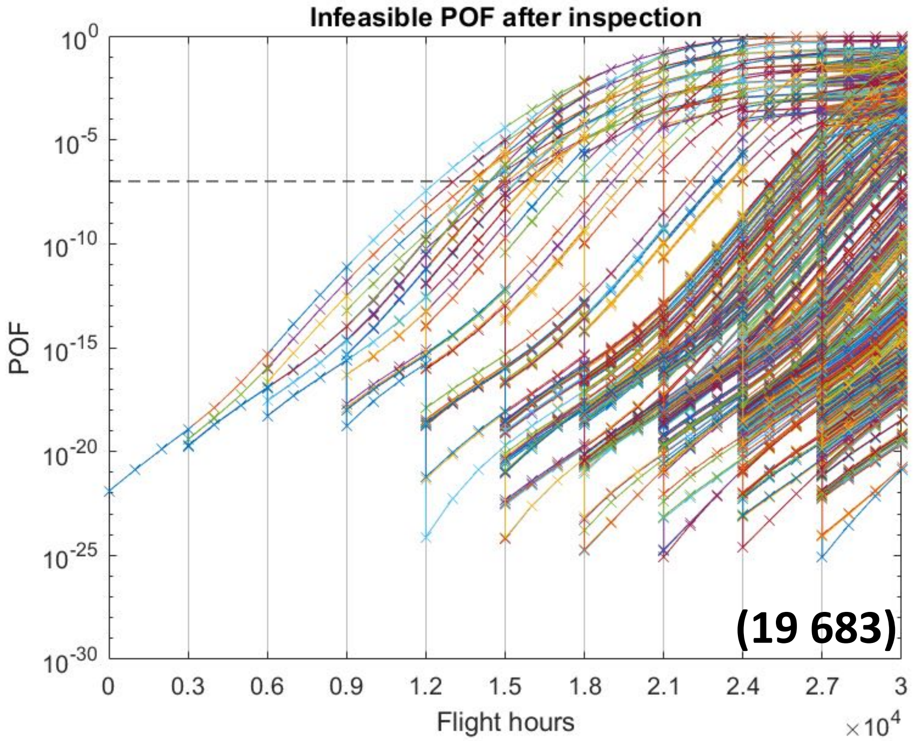
Schedule times (10^3 flight hours)

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



(19 683)

Possible inspection times [3000:3000:27000]

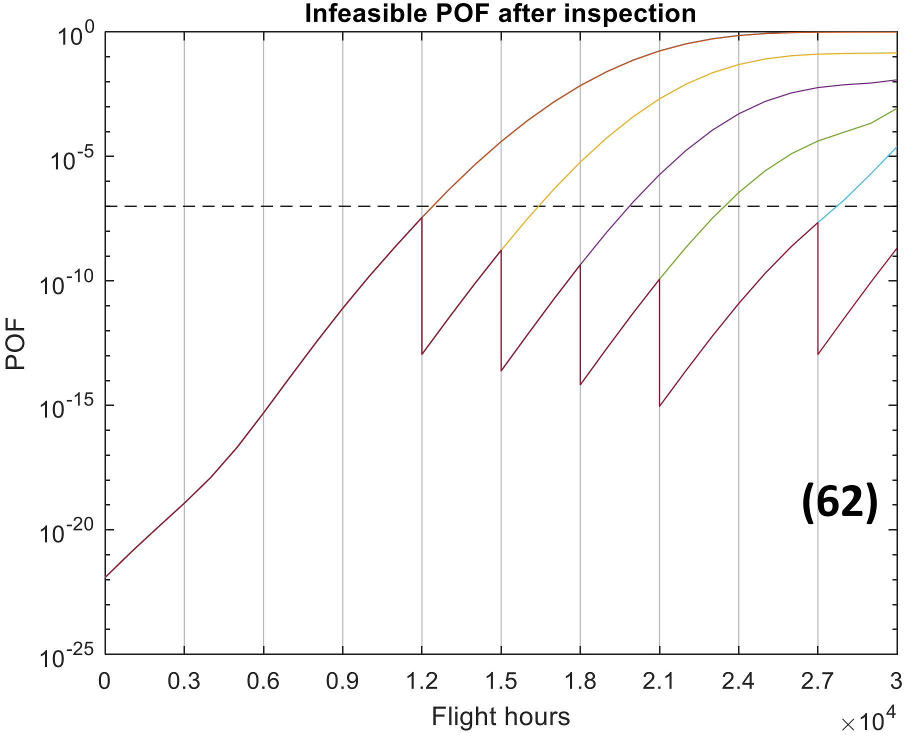
Schedule times (10^3 flight hours)

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

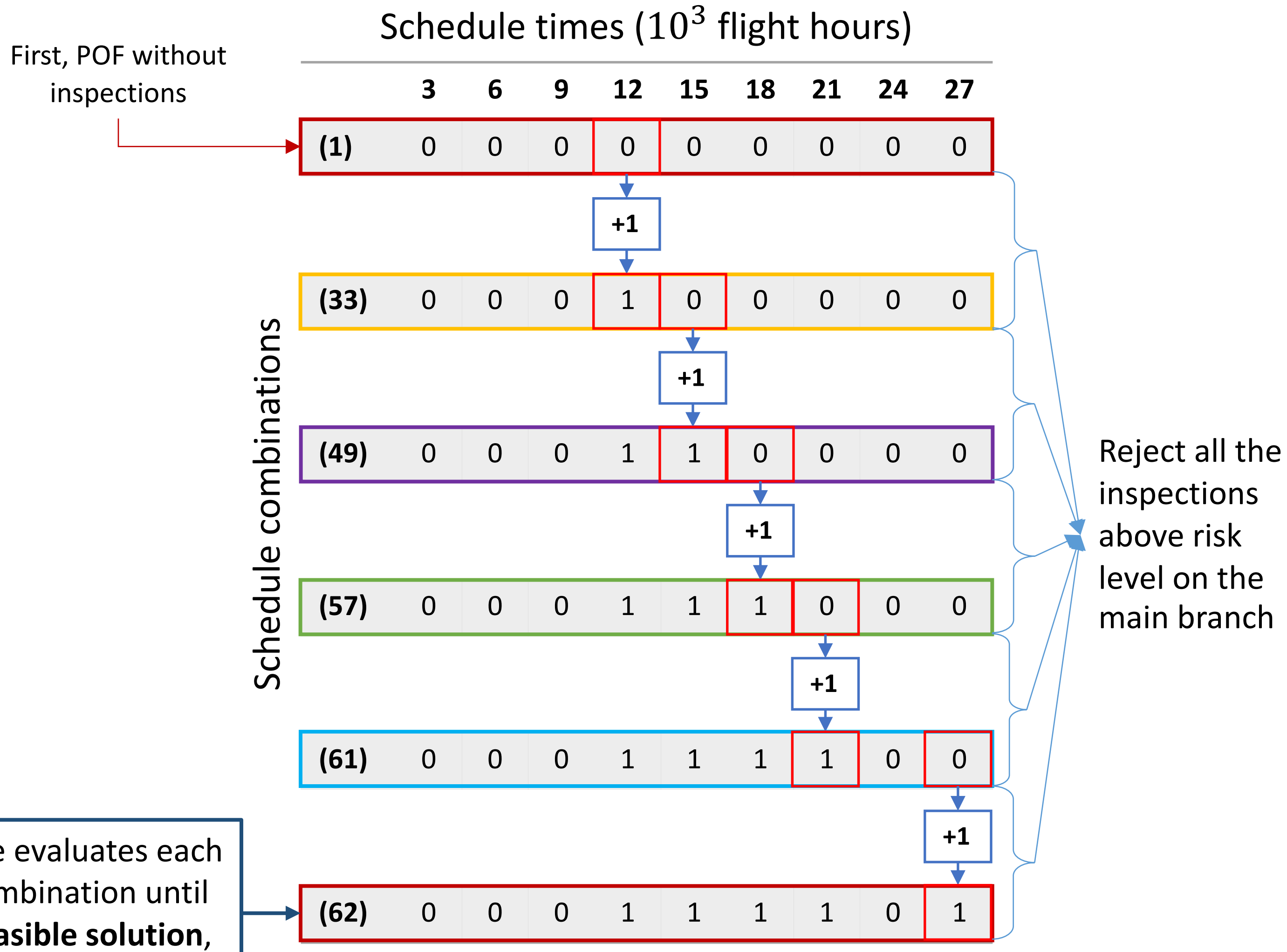
Number of possible inspections times: number of positions that will be fill with all the numerical combinations in base 3

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)

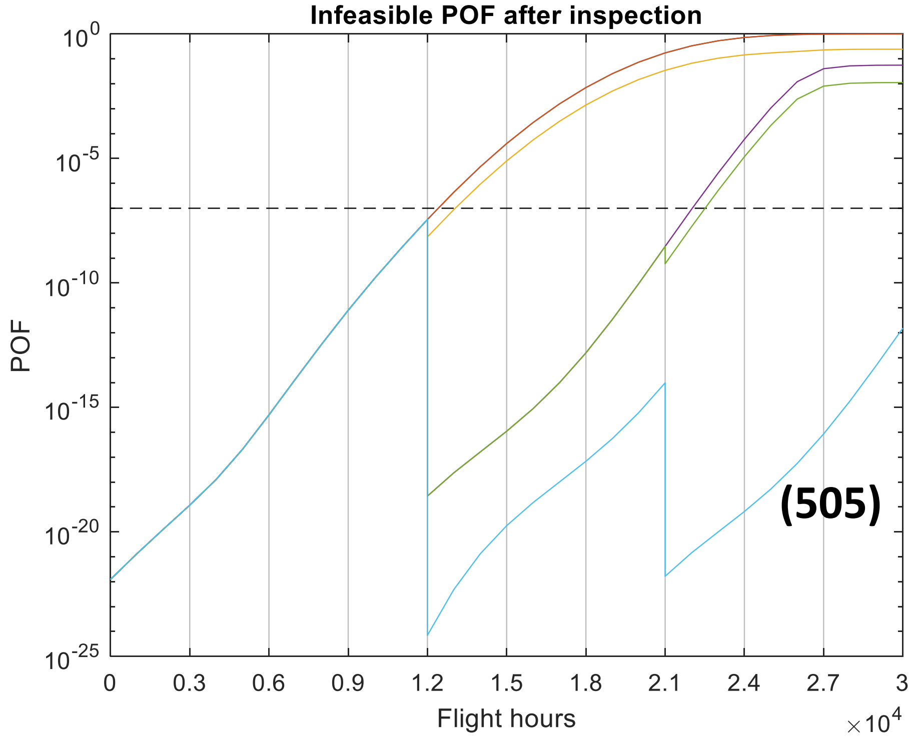
Schedule combinations

	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

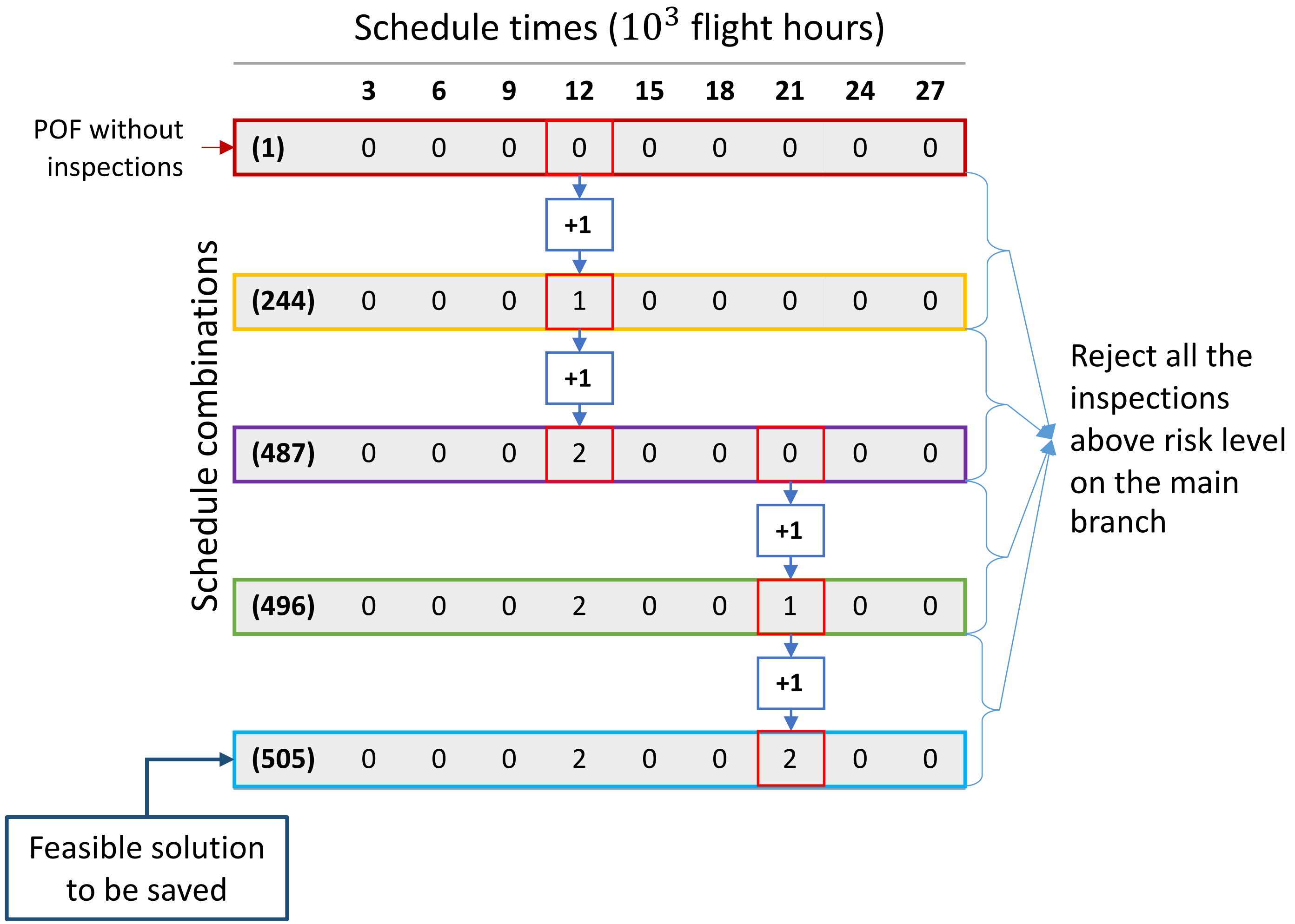
Reject all the inspections above risk level on the main branch

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]



Schedule times (10^3 flight hours)

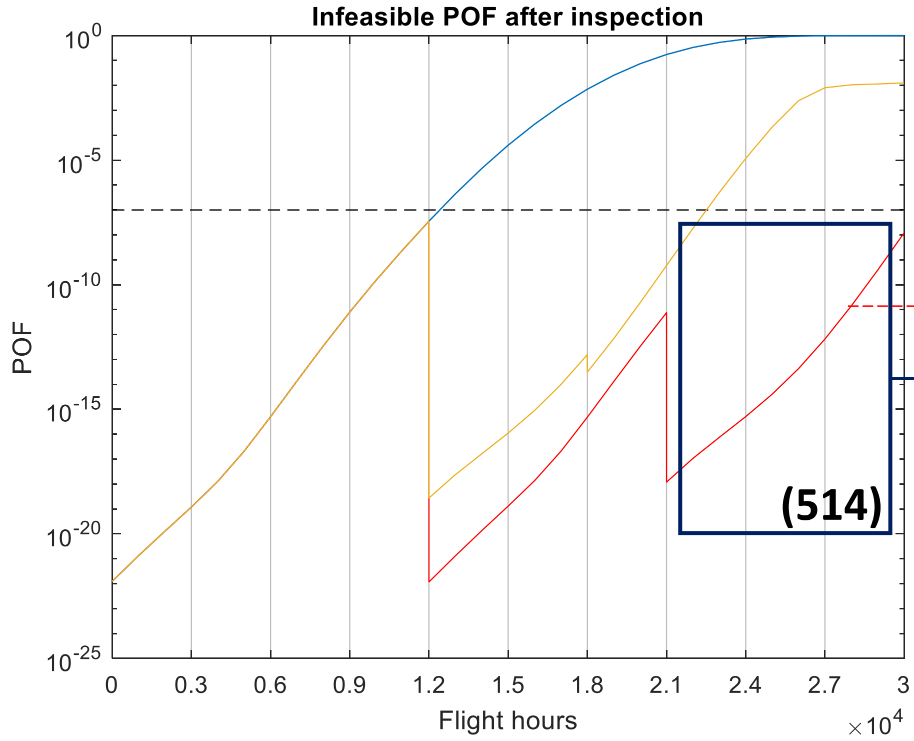
Schedule combinations

	3	6	9	12	15	18	21	24	27
(1)	0	0	0	0	0	0	0	0	0
(244)	0	0	0	1	0	0	0	0	0
(487)	0	0	0	2	0	0	0	0	0
(496)	0	0	0	2	0	0	1	0	0
(505)	0	0	0	2	0	0	2	0	0

Reject all the inspections above risk level on the main branch

Feasible solution to be saved

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)

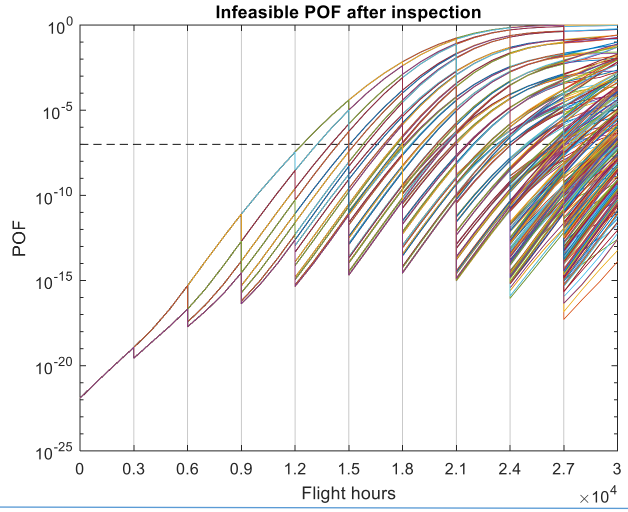
Schedule combinations	3	6	9	12	15	18	21	24	27
(1)	0	0	0	0	0	0	0	0	0
(505)	0	0	0	2	0	0	2	0	0
(506)	0	0	0	2	0	0	2	0	1
(507)	0	0	0	2	0	0	2	0	2
(508)	0	0	0	2	0	0	2	1	0
(509)	0	0	0	2	0	0	2	1	1
(510)	0	0	0	2	0	0	2	1	2
(511)	0	0	0	2	0	0	2	2	0
(512)	0	0	0	2	0	0	2	2	1
(513)	0	0	0	2	0	0	2	2	2
(514)	0	0	0	2	0	1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

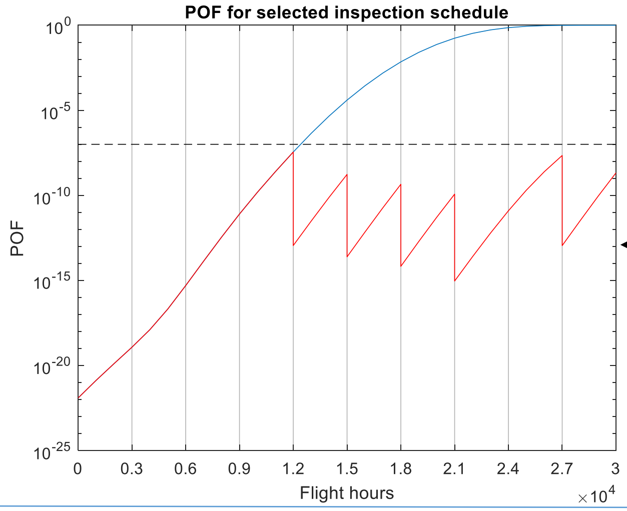
Possible inspection times [3000:3000:27000]

Method will skip POF evaluations from 505 to 514

Skipping Algorithm Validation
Single Inspection type

All combinations evaluated

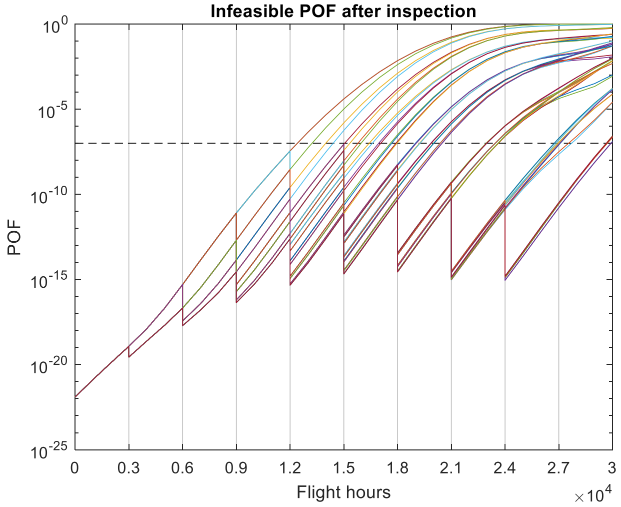

Infeasible POF after inspection

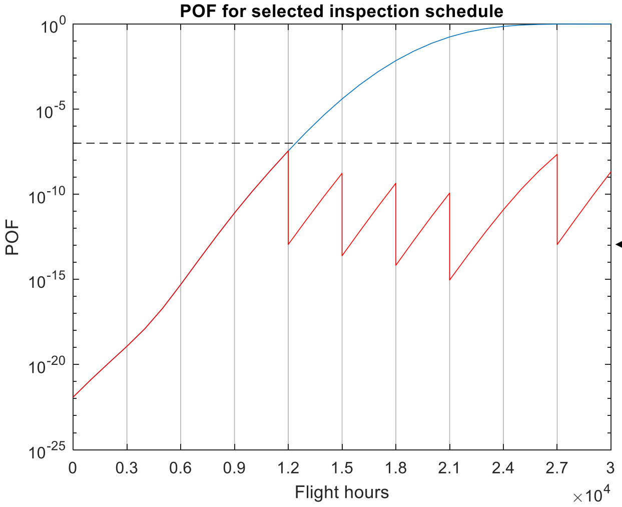

POF for selected inspection schedule

Possible inspection times [3000:3000:27000]

Same Schedule

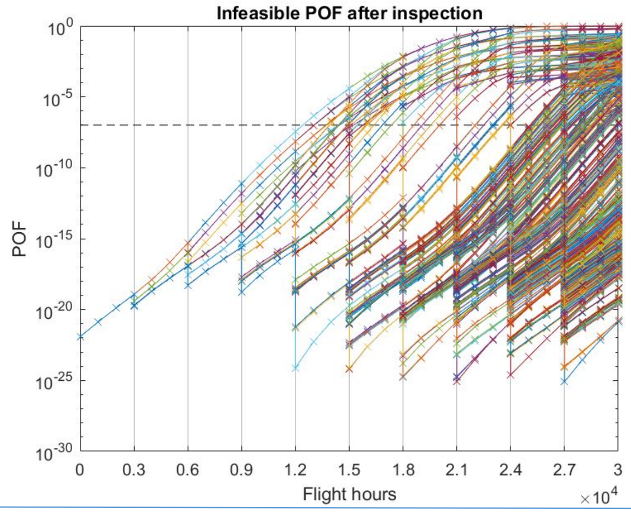
With skipping algorithm

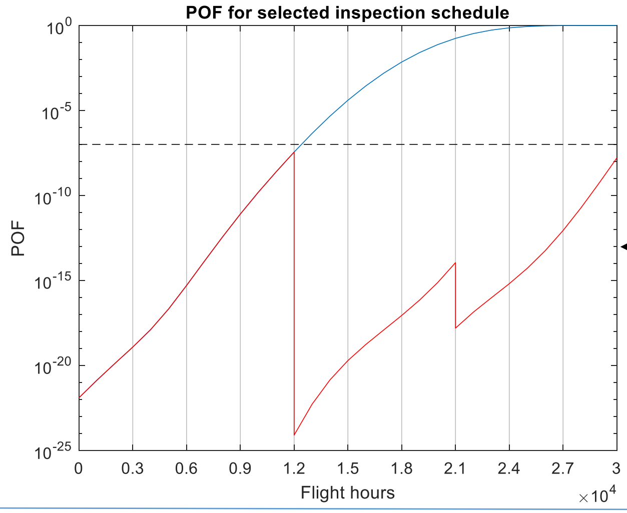

Infeasible POF after inspection


POF for selected inspection schedule

Skipping Algorithm Validation Multiple Inspection Type

All combinations evaluated

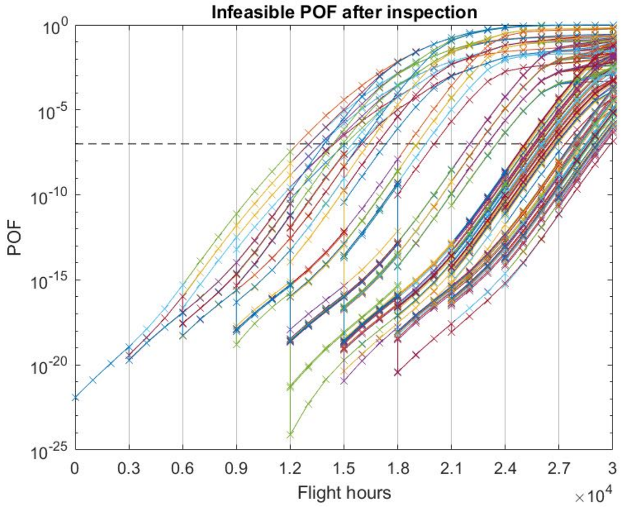


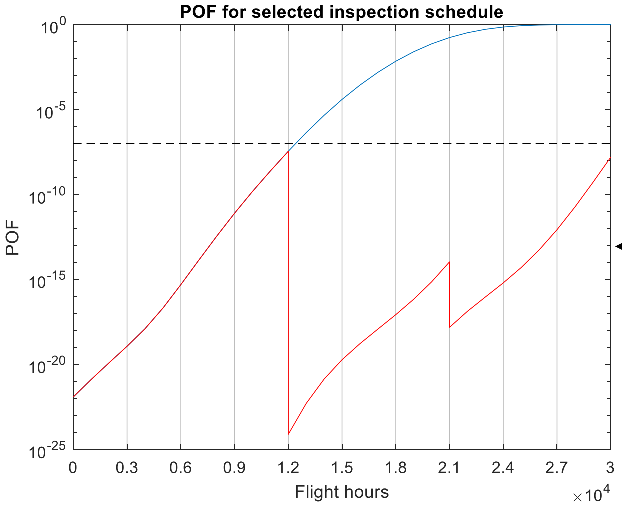


Possible inspection times [3000:3000:27000]

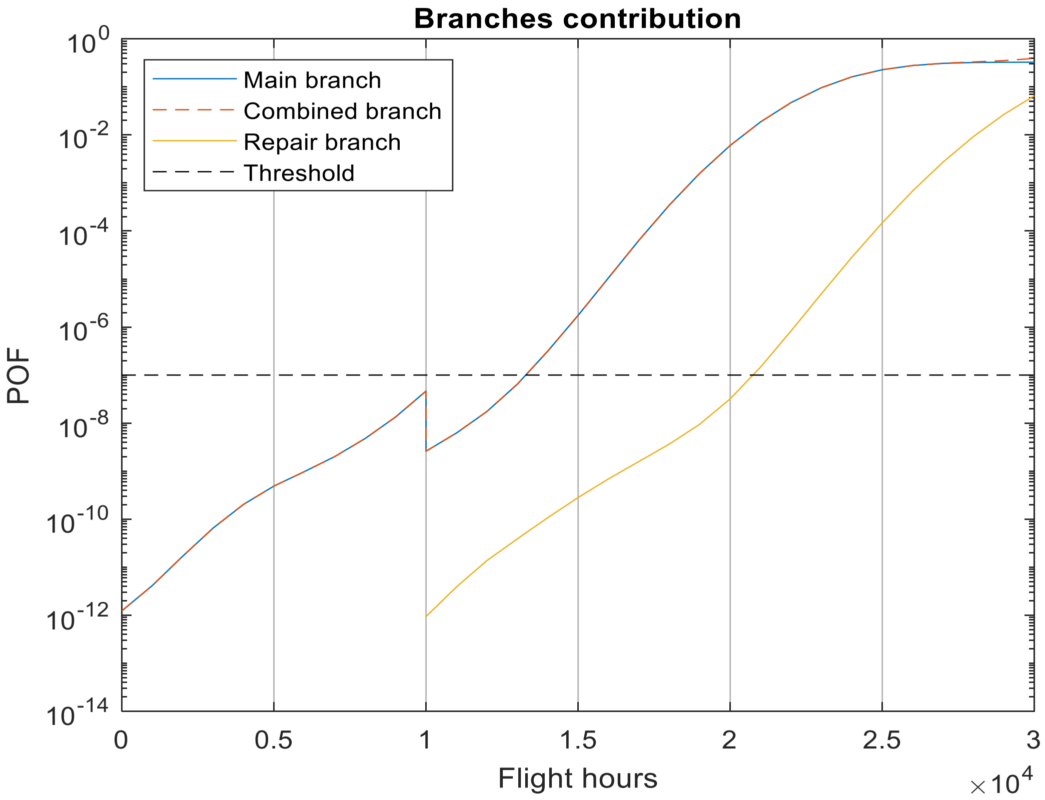
Same Schedule

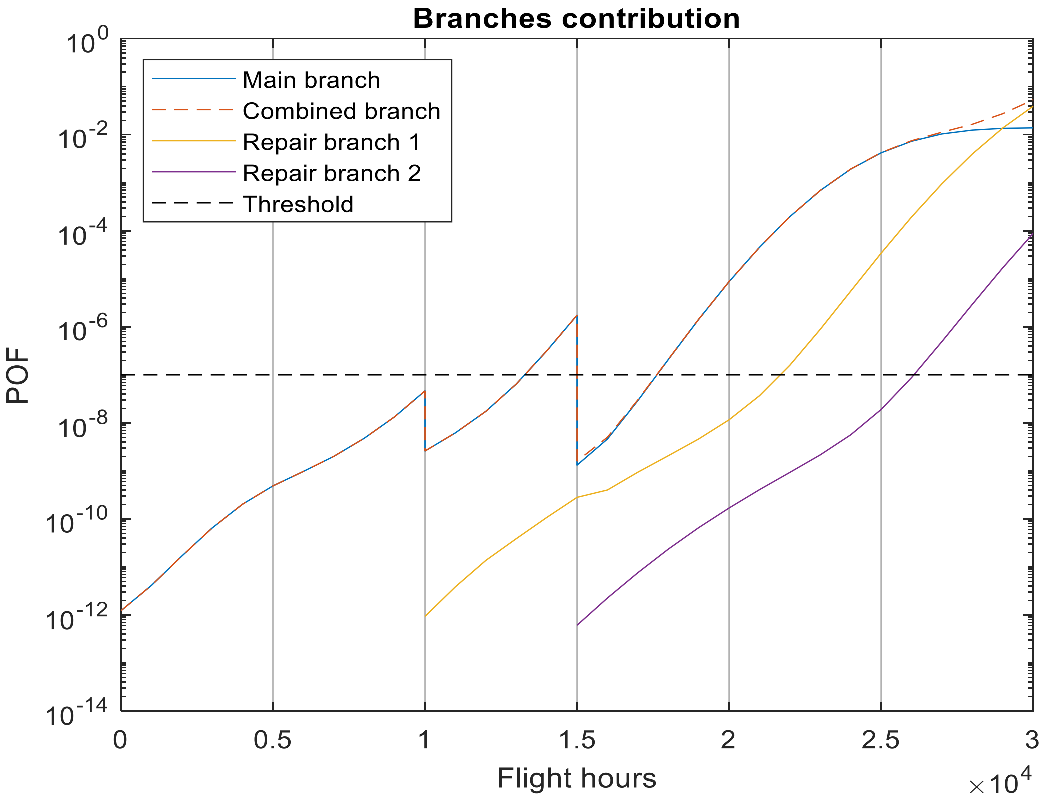
With skipping algorithm





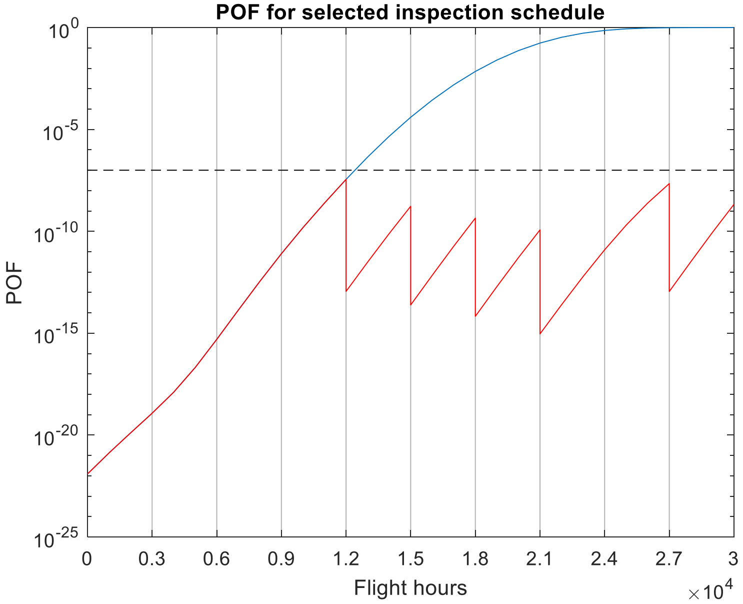
Use Main Branch Approximation

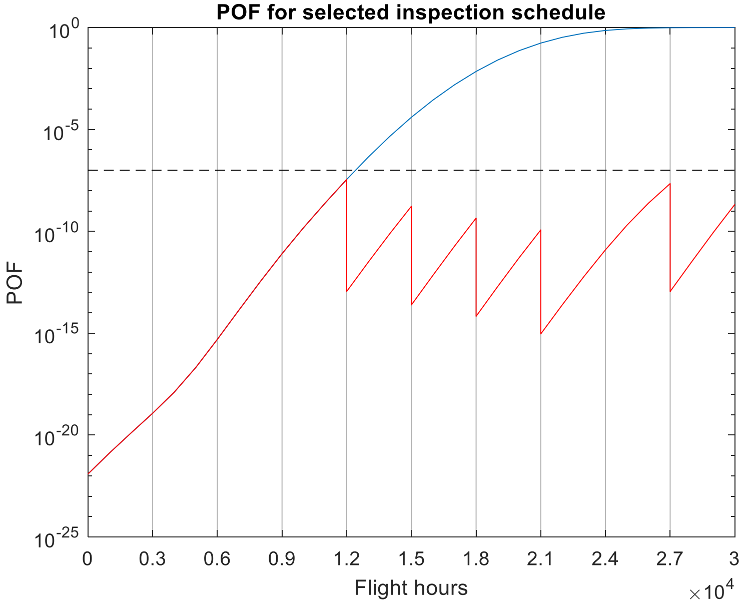


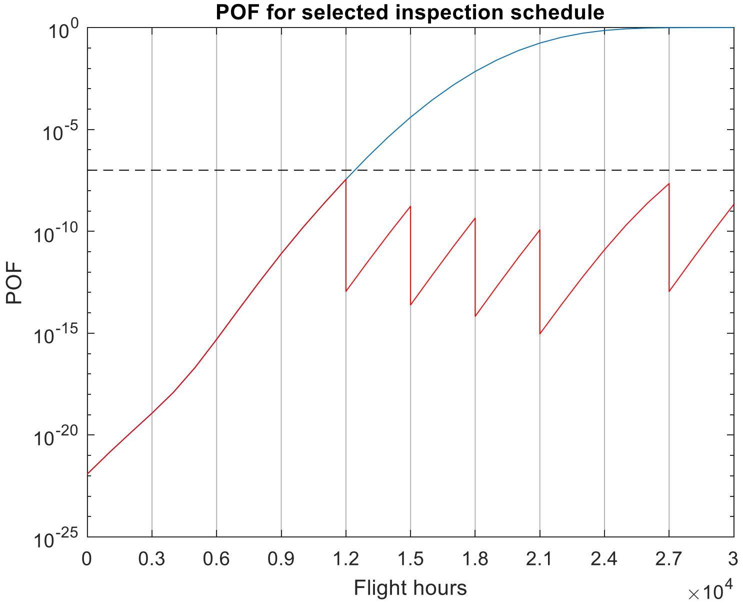


The code will only use the main branch curve information

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

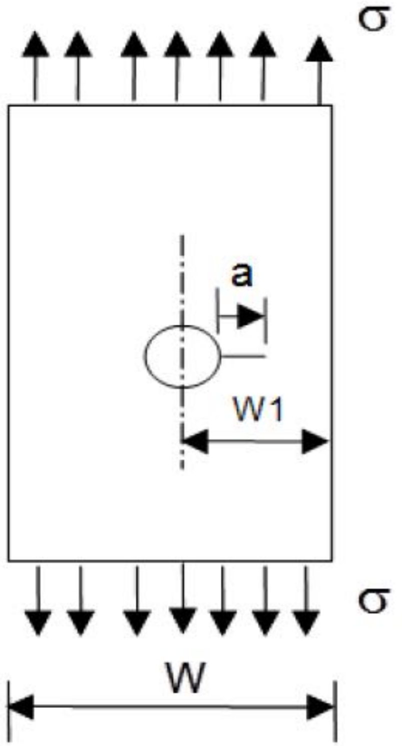
Calculation relative time: 1.7

With skipping algorithm and using main branch approximation

Calculation relative time: 1

Example

Input Data (I)



r = radius of hole

W1 = W/2

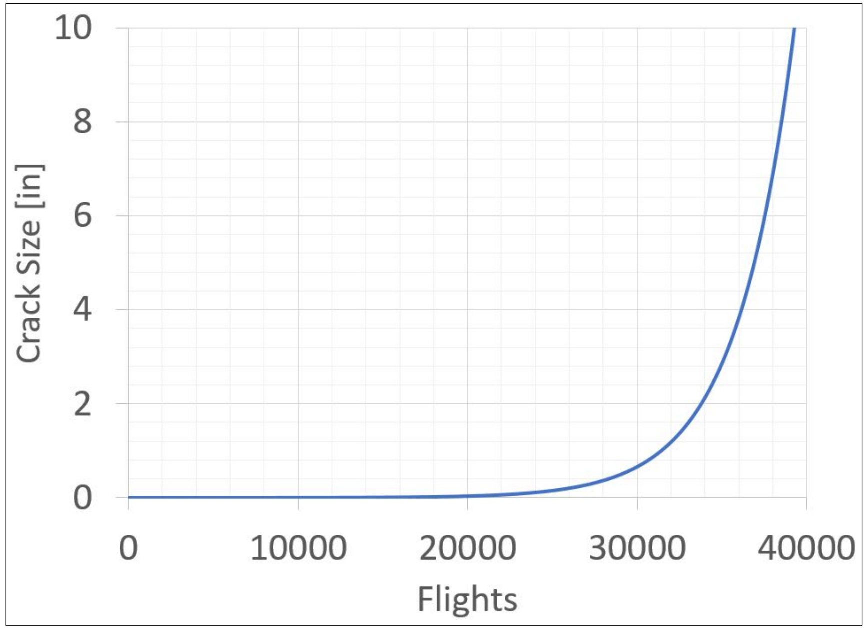
$$K = \sigma \cdot \beta \sqrt{\pi a} = \sigma \cdot \alpha$$

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

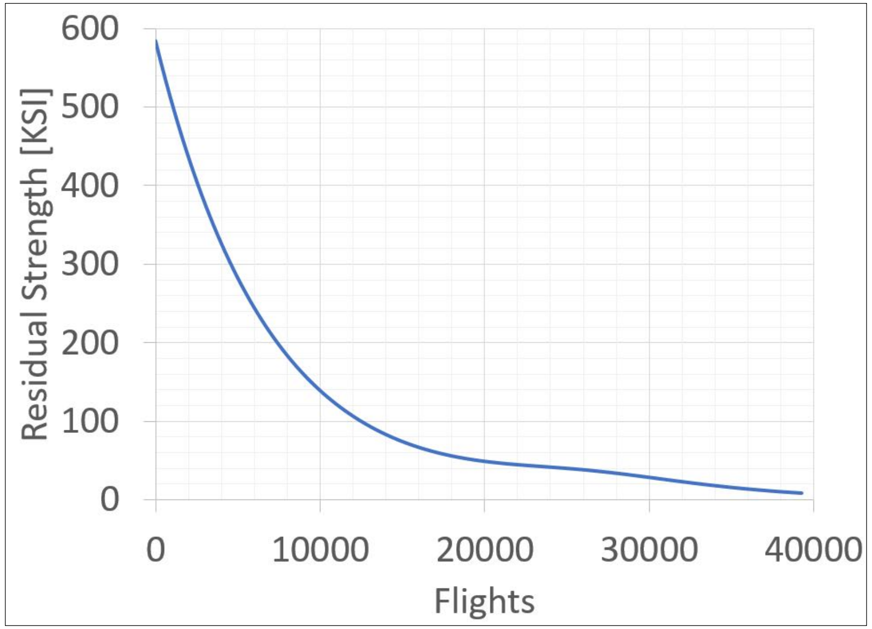
$$\beta_{hole} = 0.6762 + 0.8734 / (0.3246 + (a / R))$$

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

Crack Growth



Residual Strength

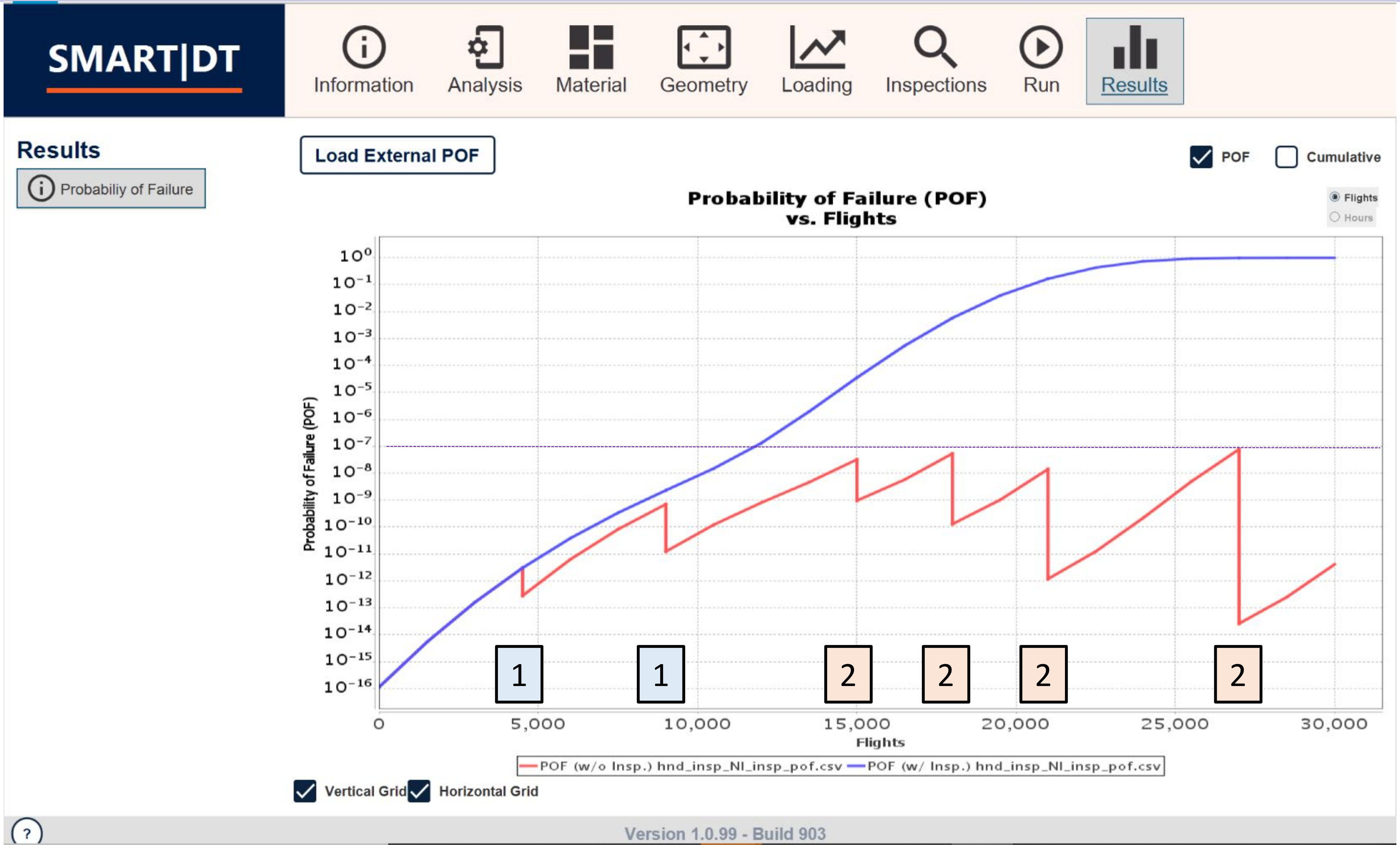


Input Data (II)

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

● Flights ○ Hours

Probability of Failure (POF) vs. Flights

🟥 POF (w/o Insp.) hnd_insp_NI_insp_pof.csv 🟦 POF (w/ Insp.) hnd_insp_NI_insp_pof.csv

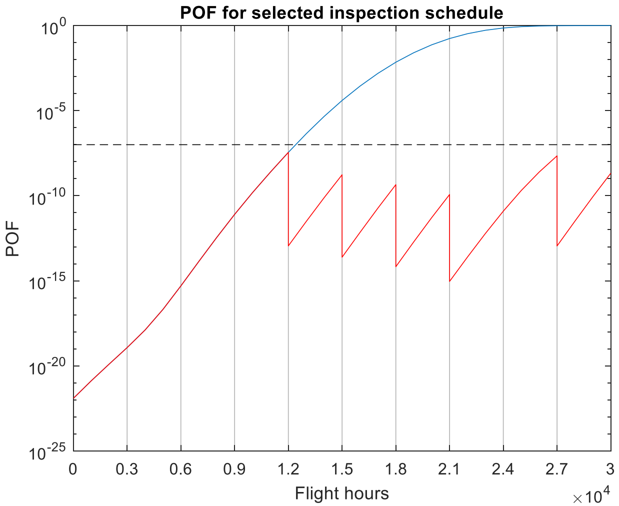
☑ Vertical Grid ☑ Horizontal Grid

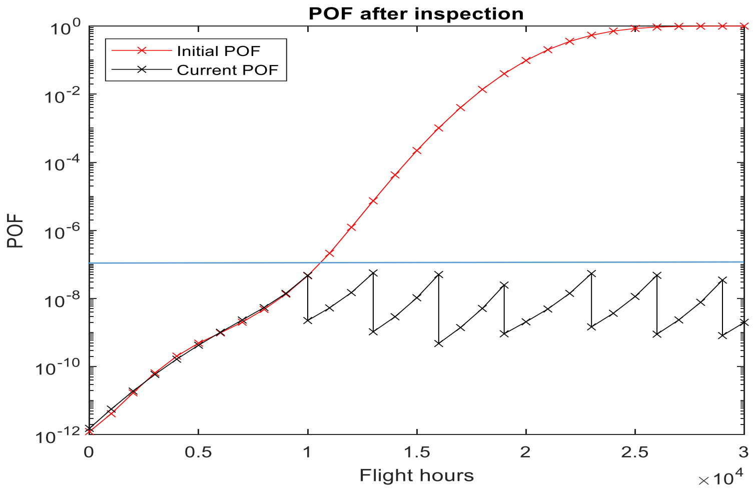
?

Version 1.0.99 - Build 903

Future Work

- Implement the Shortest Path Method (SPM) in SMART|DT.
- 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