Pointing Error Engineering Tool (PEET)

Final Presentation

ESTEC Frame Contract 19179/05/NL/LvH Work Order 19

ESA/ESTEC

November 15, 2012

Agenda

Morning Session

- Introduction to the Pointing Error Engineering Handbook
- Project overview
- Presentation of the PEET software
- Summary and status report

Afternoon (room Nc321)

- Live demonstration of PEET
- Questions & answers session

Pointing Error Engineering Tool

These are the main objectives of PEET:

- PEET shall support the ESA PEE Handbook users in compiling pointing error budgets
- PEET shall support interactive breakdown and budgeting with the possibility for the user to visually examine the response to his actions
- PEET shall assist the user by providing a database of common pointing error sources

Work Packages

WP1 Requirements Capture

WP2 PointingSat Reference Case Definition

WP3 Software Core Design

WP4 GUI Design

WP5 Database Design

WP6 Testing and Validation

Work started mid of January 2012

PEET Prototype Development Team

PEET Prototype Development

Astos Solutions

- Prime Contractor
- GUI Development
- Core Development
- Testing and Verification
- Project Management

Institute of Flight Mechanics and Control, University of Stuttgart

- Consultancy
- Experts in Pointing Error Engineering and GNC
- Co-authors of the Pointing Error Engineering Handbook
- Definition of the PointingSat study case

PEET Development Process

The development of PEET is divided into two phases:

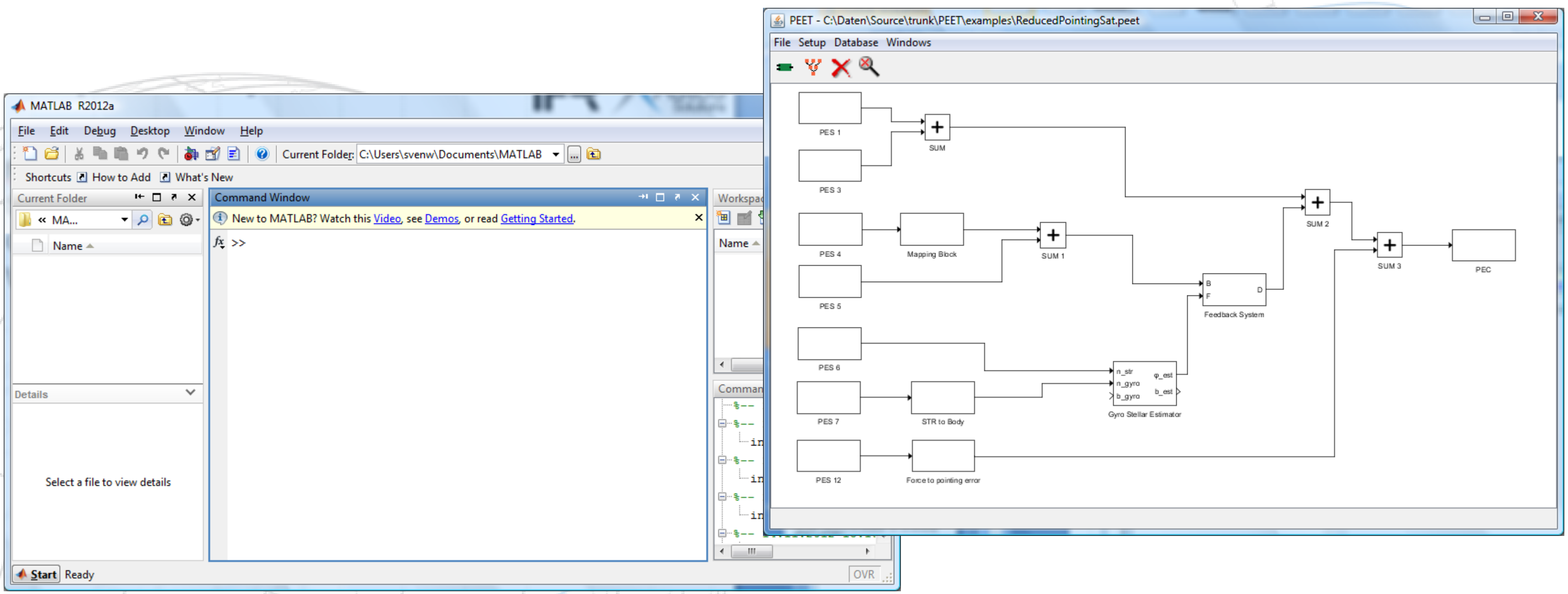
1. A prototyping phase with special focus on pre-phase A (CDF studies) and Phase A activities
2. A development phase, potentially funded by the GSTP programme, supporting as well detailed pointing error engineering tasks of Phase B and Phase C/D.

PEET Prototype Development

- The software is designed as a MATLAB toolbox
- The GUI is written in Java while the core algorithms are written in MATLAB, thus they can reuse built-in MATLAB functions
- Software documentation according to ECSS standards (SRD, ICD, SDD, SUM, SVP) is available
- Source code is version controlled using Subversion VCS
- Continuous integration/testing using Hudson CI
- Bug tracking using Request Tracker

Prerequisites / Dependencies

- MATLAB (>2009a), 32 bit version
- Simulink
- Control System Toolbox
- Windows Operating System (tested with XP, Vista and 7)



PEET License Conditions

PEET will be published under the „ESA Community Open License“:

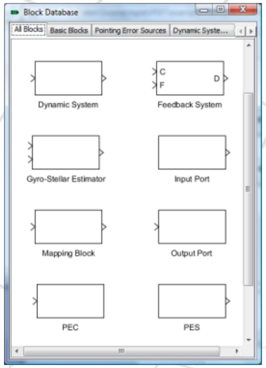
- Open source for ESA member countries
- License-free access to PEET for ESA member countries.

➡ PEET can/shall be used by European industry (e.g. within ESA projects)

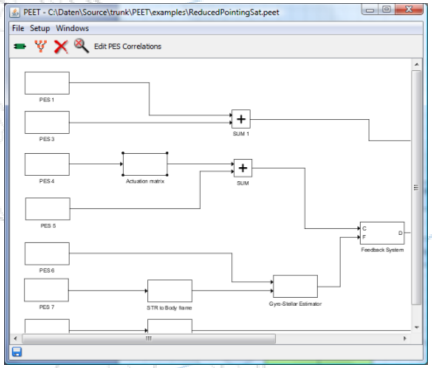
PEET Workflow

The PEET workflow is fully compatible with the methodology defined in the PEEH:

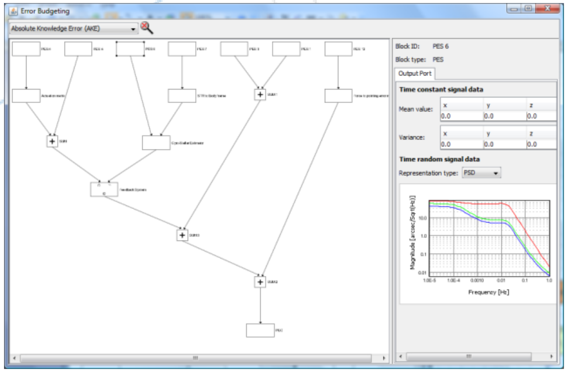
1. Characterization of pointing error sources (AST-1)
2. Setup the pointing error system within the System Editor of PEET using blocks from the library (AST-2)
3. Analyse the pointing error budget or do break down studies using the Tree View of PEET (AST-3 & AST-4)



Block Database

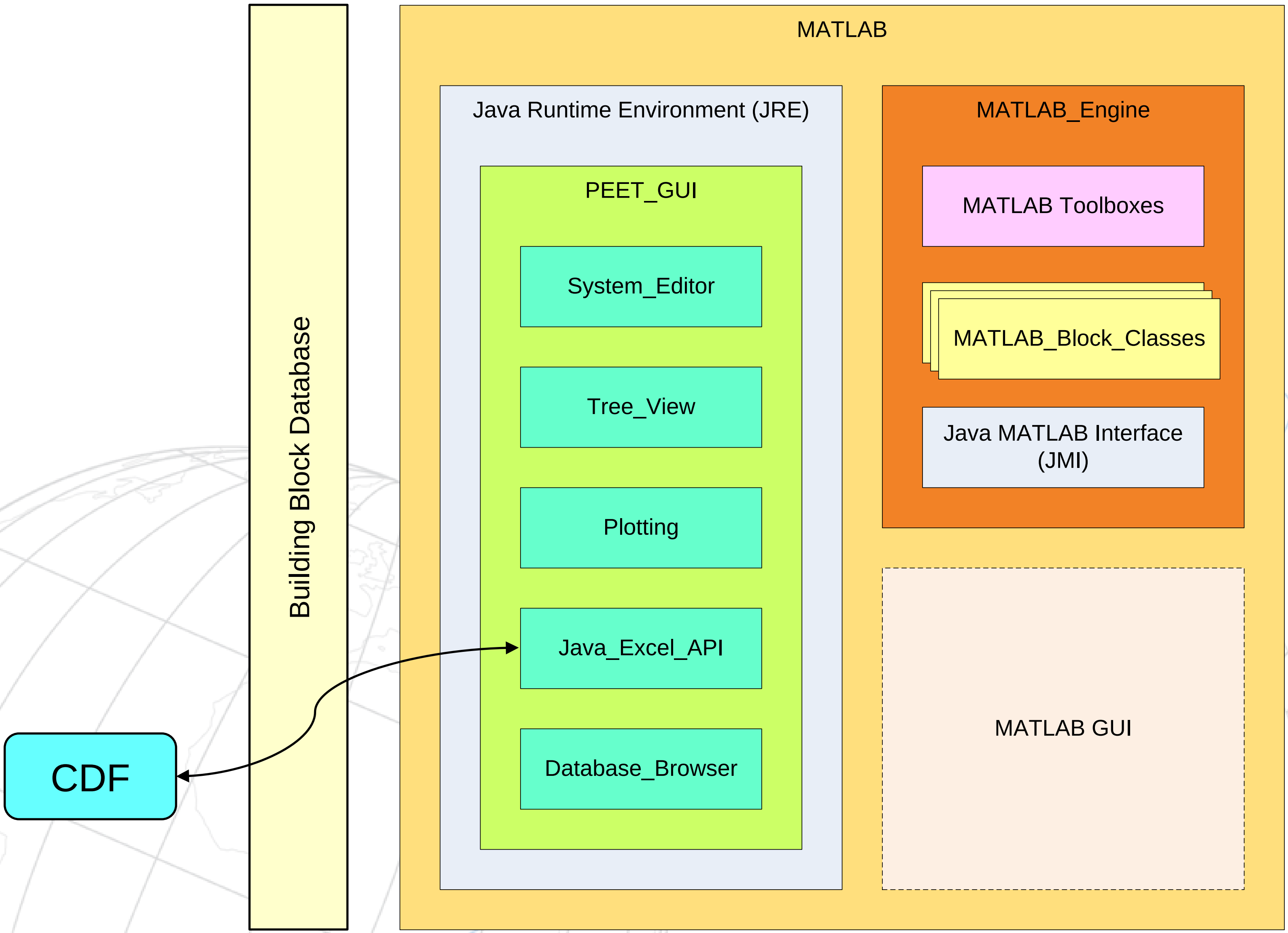


System Editor

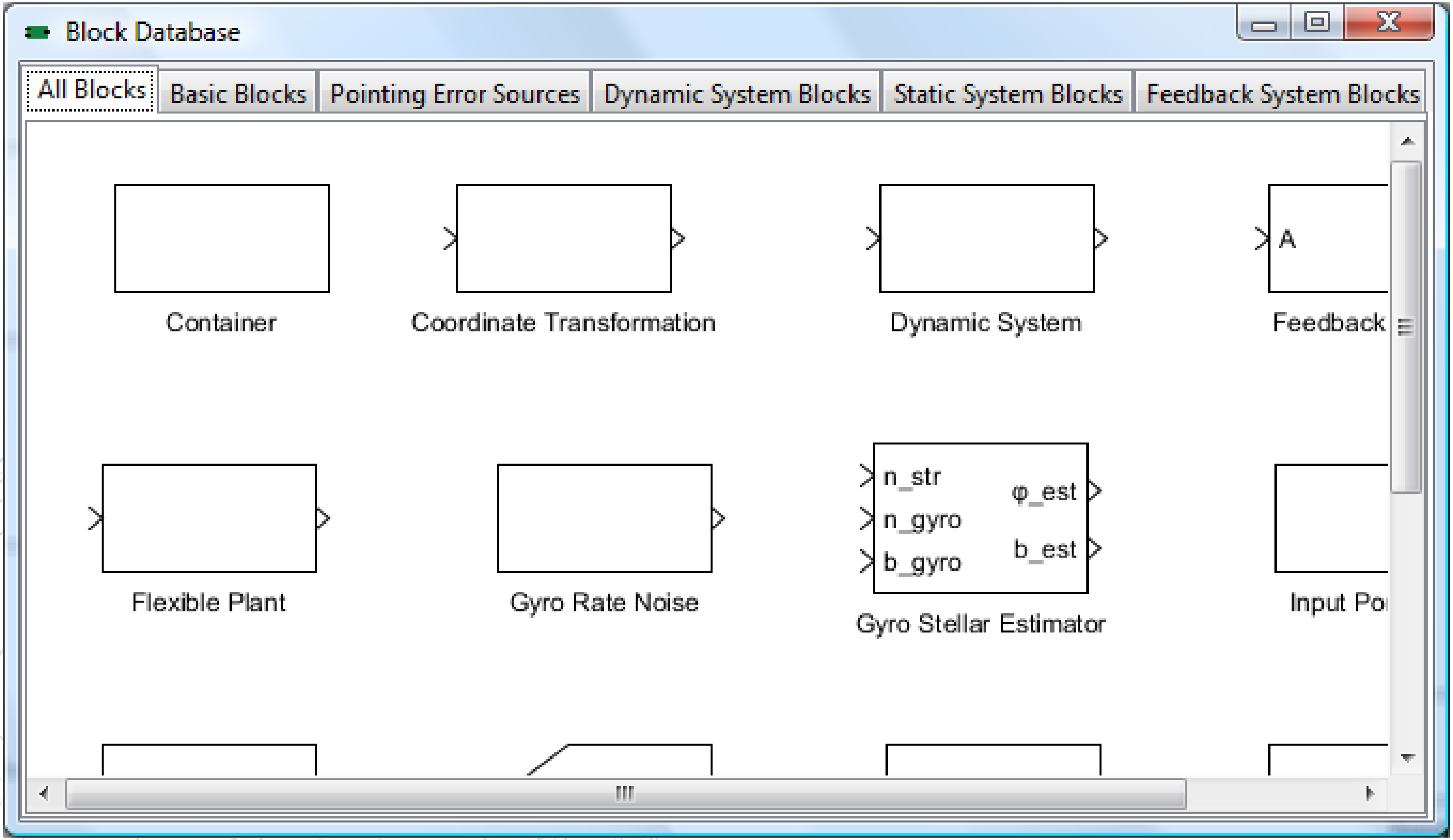


Tree View

PEET Architecture



PEET Building Blocks

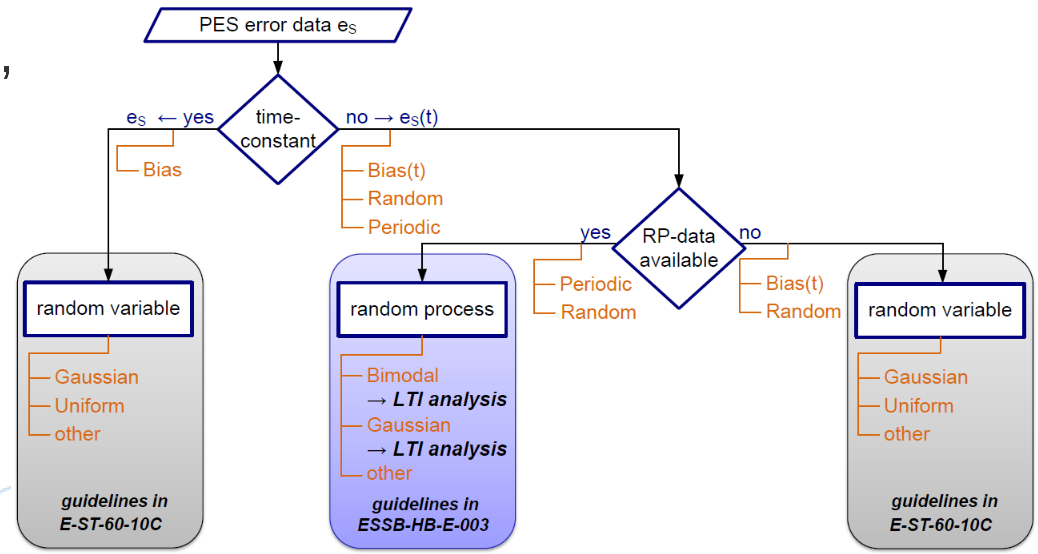


AST-1

Characterization of PES

PEET Pointing Error Sources

- 1- or 3-dimensional pointing error sources
- Time-constant
  - Uniform, Gaussian, bimodal, Rayleigh, discrete
- Time-random
  - Random variables
    - Uniform, Gaussian, drift
  - Random processes
    - Time series
    - Power spectral density
    - Covariance matrix
    - Periodic process (defined by frequency and amplitude)
- Time-constant and time-random parts can be both defined in parallel



Examples of PES (AST-1)

Typical pointing error sources that are part of the PointingSat reference case:

Payload-star tracker alignment knowledge (assembly)
Payload-star tracker alignment knowledge (launch)
Thruster noise
Temporal star tracker noise
Gyro noise
Environmental disturbances
Cryocooler micro-vibrations

Pointing Error Sources (AST-1)

Description of PES:

- User defines signal class and distribution type (for random description).
- PEET automatically performs statistical interpretation according to user-selection in line with PEEH or ECSS tables
- Example: temporal Gaussian distribution with uniformly distributed variance (e.g. T-dependence)

$$\alpha = G(0, U(\sigma^2_{min}, \sigma^2_{max}))$$

$$\alpha_{temporal} = G(0, \sigma^2_{max})$$

$$\alpha_{ensemble} = \frac{1}{3} U(\sigma^2_{min}, \sigma^2_{max})$$

$$\alpha_{mixed} = \int G(0, \sigma^2)\, U(\sigma^2_{min}, \sigma^2_{max})\, d\sigma$$

Time-Constant PES

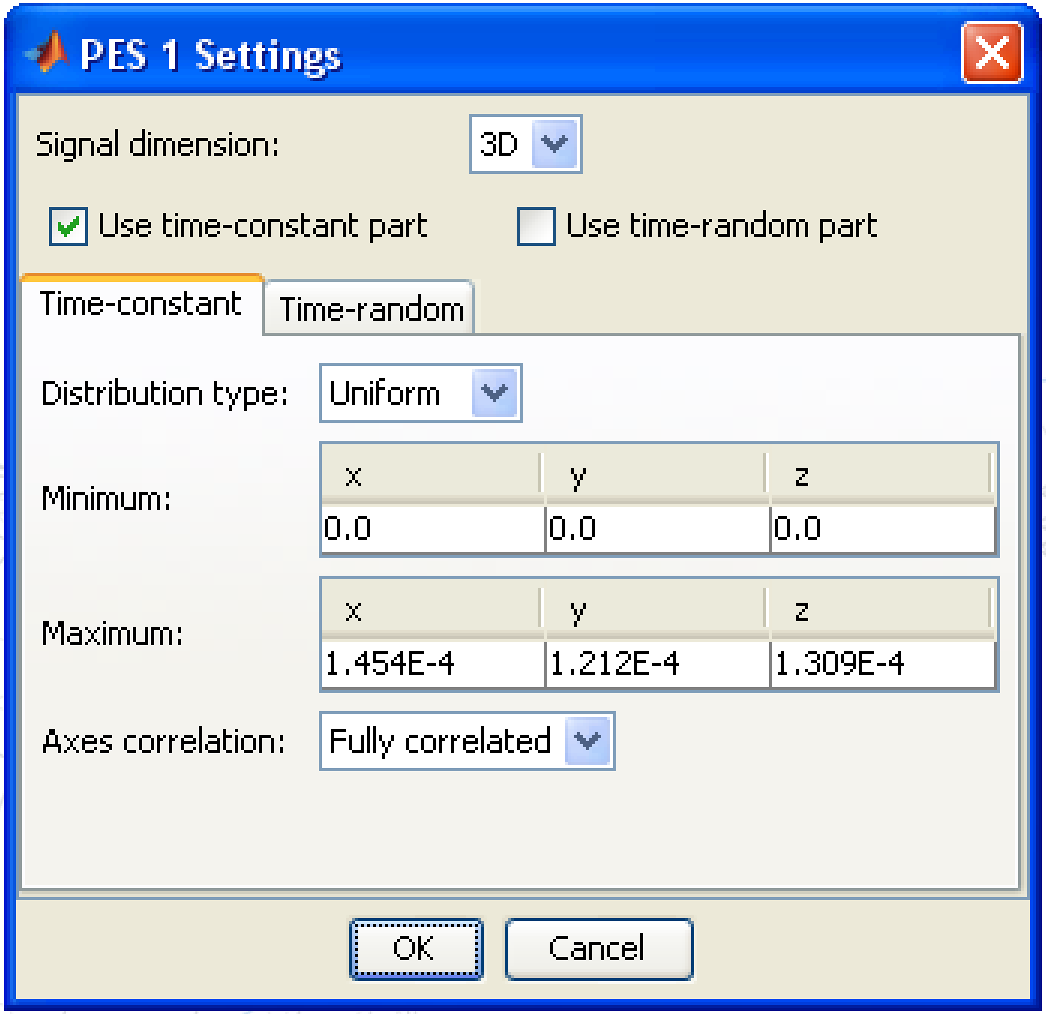
Example: Payload-STR misalignment knowledge (integration)

- Constant after assembly
- Based on theodolite measurements

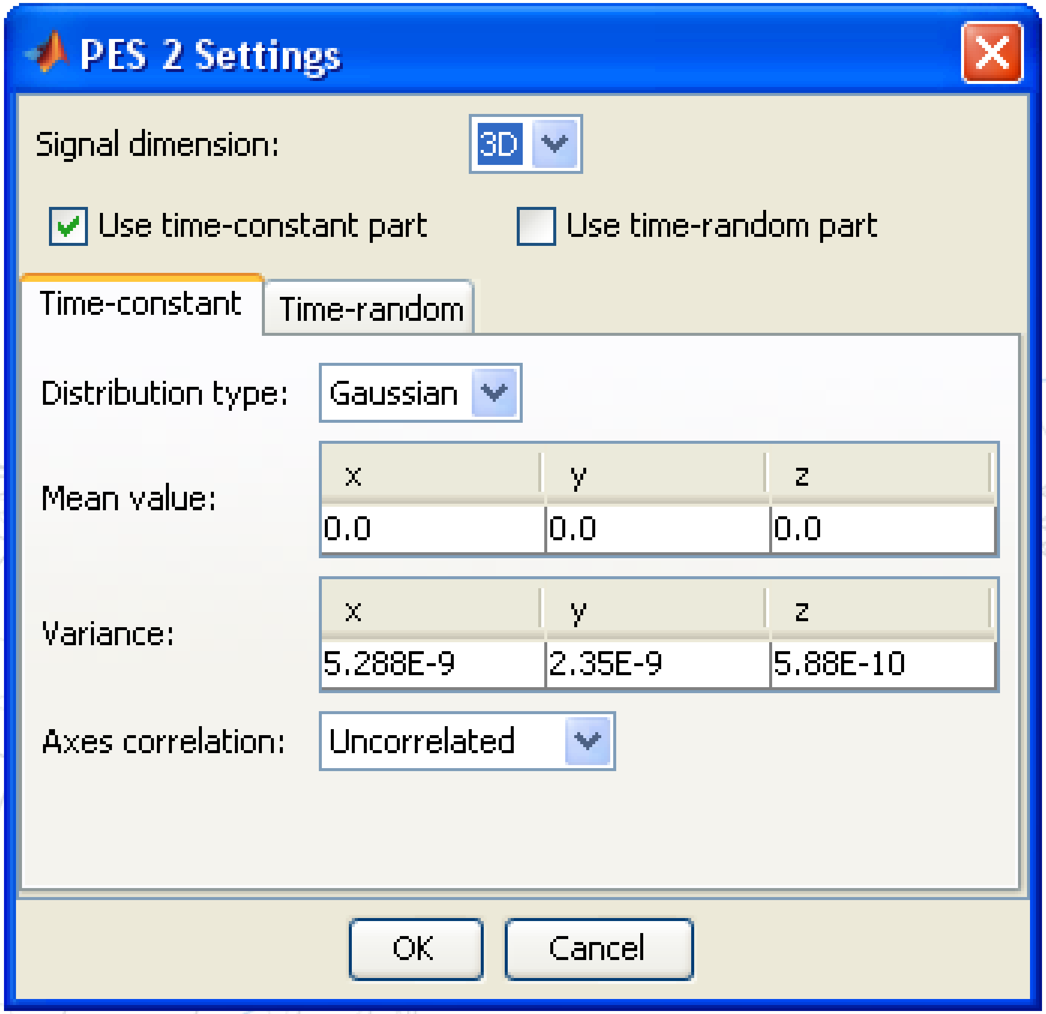
➡ Time-constant random variable with uniform distribution
➡ Fully correlated among different axes
➡ No correlation with other PES

$$\boldsymbol{\alpha}_1 = \begin{bmatrix} U(0, \alpha_{1,max,x}) \\ U(0, \alpha_{1,max,y}) \\ U(0, \alpha_{1,max,z}) \end{bmatrix} \text{arcsec} = \begin{bmatrix} U(0, 27) \\ U(0, 25) \\ U(0, 30) \end{bmatrix} \text{arcsec}$$

PES Dialog for Time-constant PESs



PES Dialog for Time-constant PESs

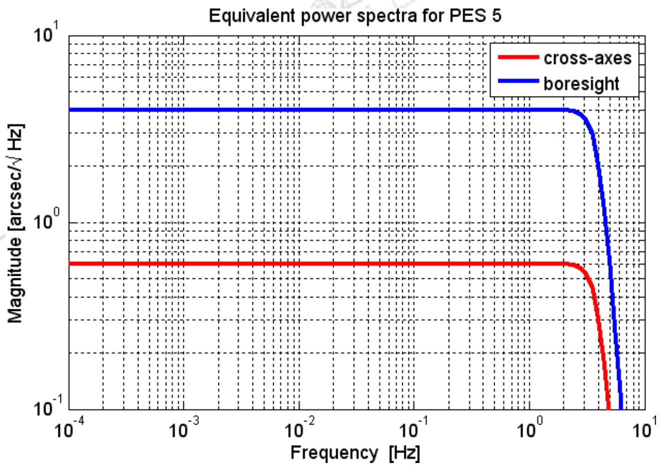


Random Process PES (AST-1)

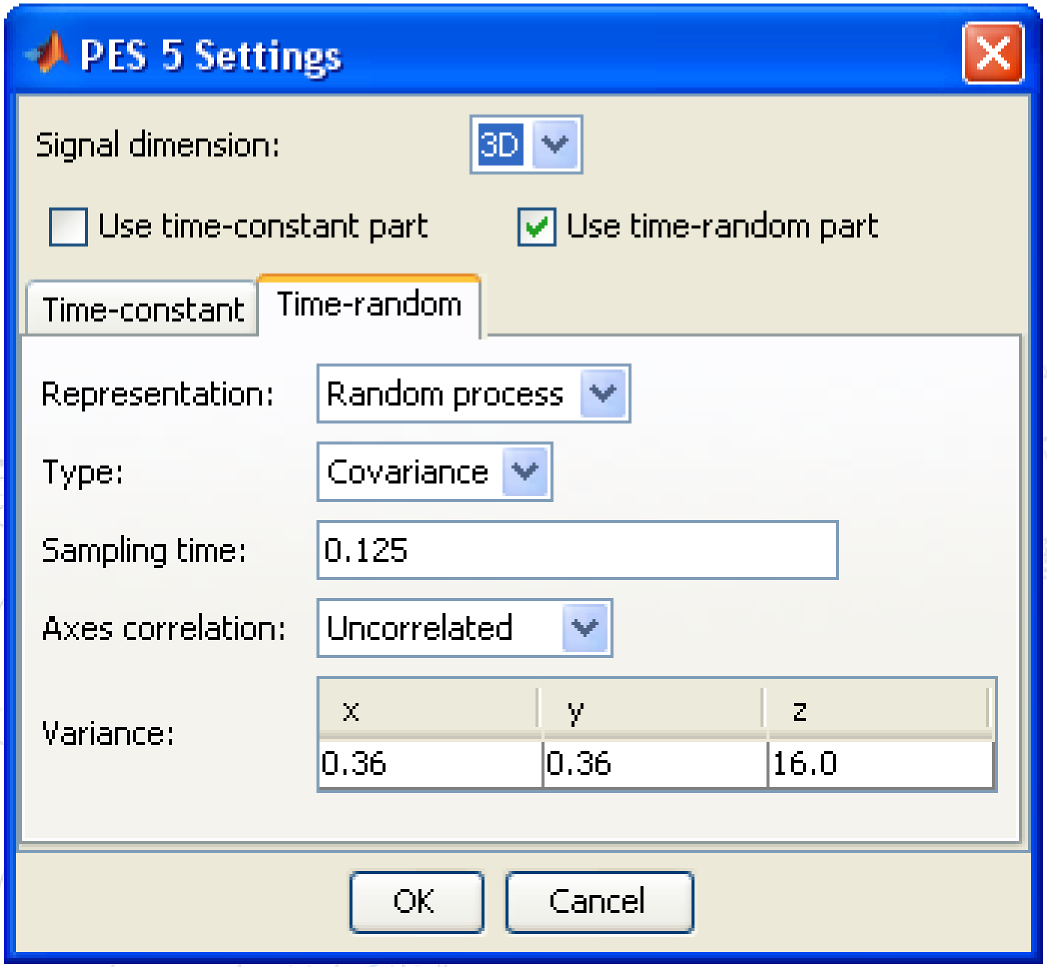
Example: Temporal star tracker noise

- Given by variances on cross-boresight and boresight axis together with sensor sampling time
- ➡ Random process described as covariance (zero-diagonals)
- ➡ Automatically converted to PSD with respective variance using low-pass filter (cut-off at Nyquist frequency)

$$\boldsymbol{\alpha}_5 = \begin{bmatrix} \sigma^2_{5,c} \\ \sigma^2_{5,c} \\ \sigma^2_{5,a} \end{bmatrix} \text{arcsec} = \begin{bmatrix} 1.55^2 \\ 1.55^2 \\ 4^2 \end{bmatrix} \text{arcsec}$$



Random Processes



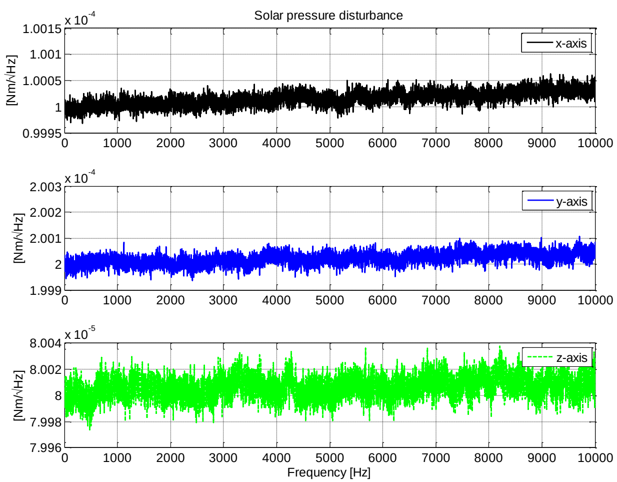
Time Series PES (AST-1)

Example: Environmental disturbances

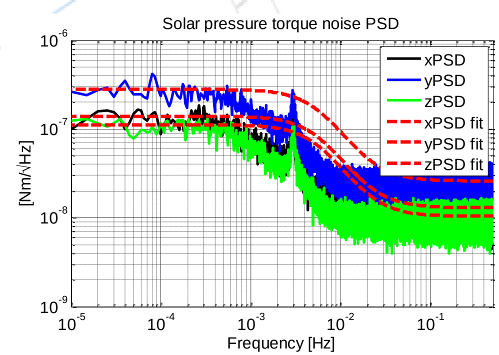
- Time series of environmental noise (mainly solar pressure) from simulation (imported from file)

➡ Automatic conversion into PSD, Bias and drift type PES

➡ PEET derives full correlation state (cross-spectral densities)

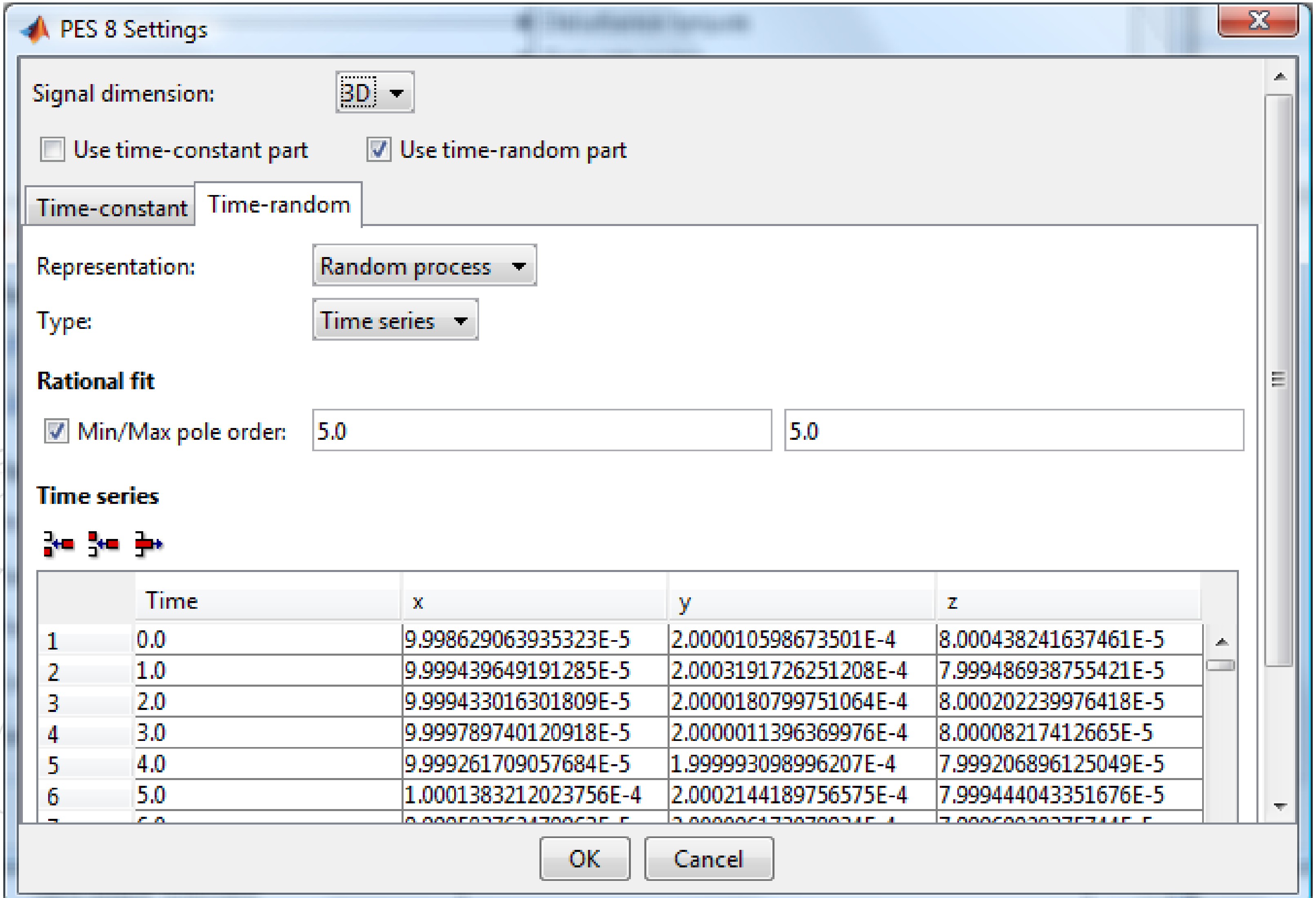


$$\boldsymbol{\varepsilon}_{D,9} = \begin{bmatrix} \varepsilon_D(D_{9,x},\Delta t_{9,D}) \\ \varepsilon_D(D_{9,y},\Delta t_{9,D}) \\ \varepsilon_D(D_{9,z},\Delta t_{9,D}) \end{bmatrix} Nm/s = \begin{bmatrix} \varepsilon_D(1.0\cdot 10^{-4},10^{6}s) \\ \varepsilon_D(2.0\cdot 10^{-4},10^{6}s) \\ \varepsilon_D(3.0\cdot 10^{-5},10^{6}s) \end{bmatrix} Nm/s$$



$$\mathbf{B}_9 = \begin{bmatrix} \delta(B_{9,x}) \\ \delta(B_{9,y}) \\ \delta(B_{9,z}) \end{bmatrix} Nm \begin{bmatrix} 1.0 \\ 2.0 \\ 0.8 \end{bmatrix} mNm$$

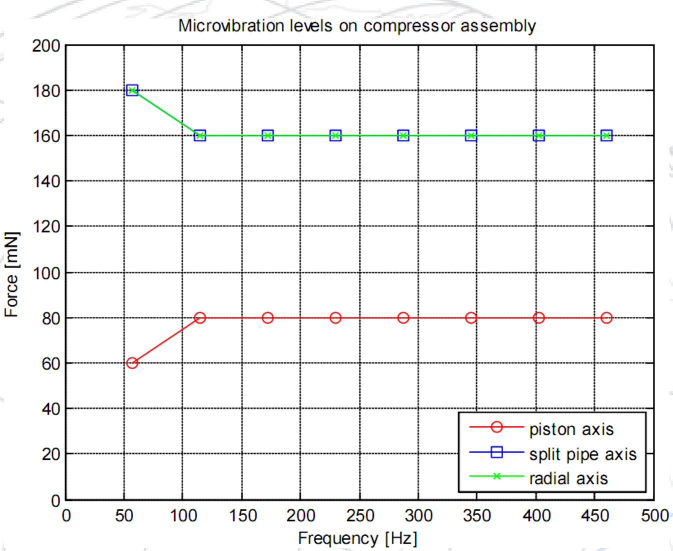
Time Series PES Dialog



Periodic Random Process PES

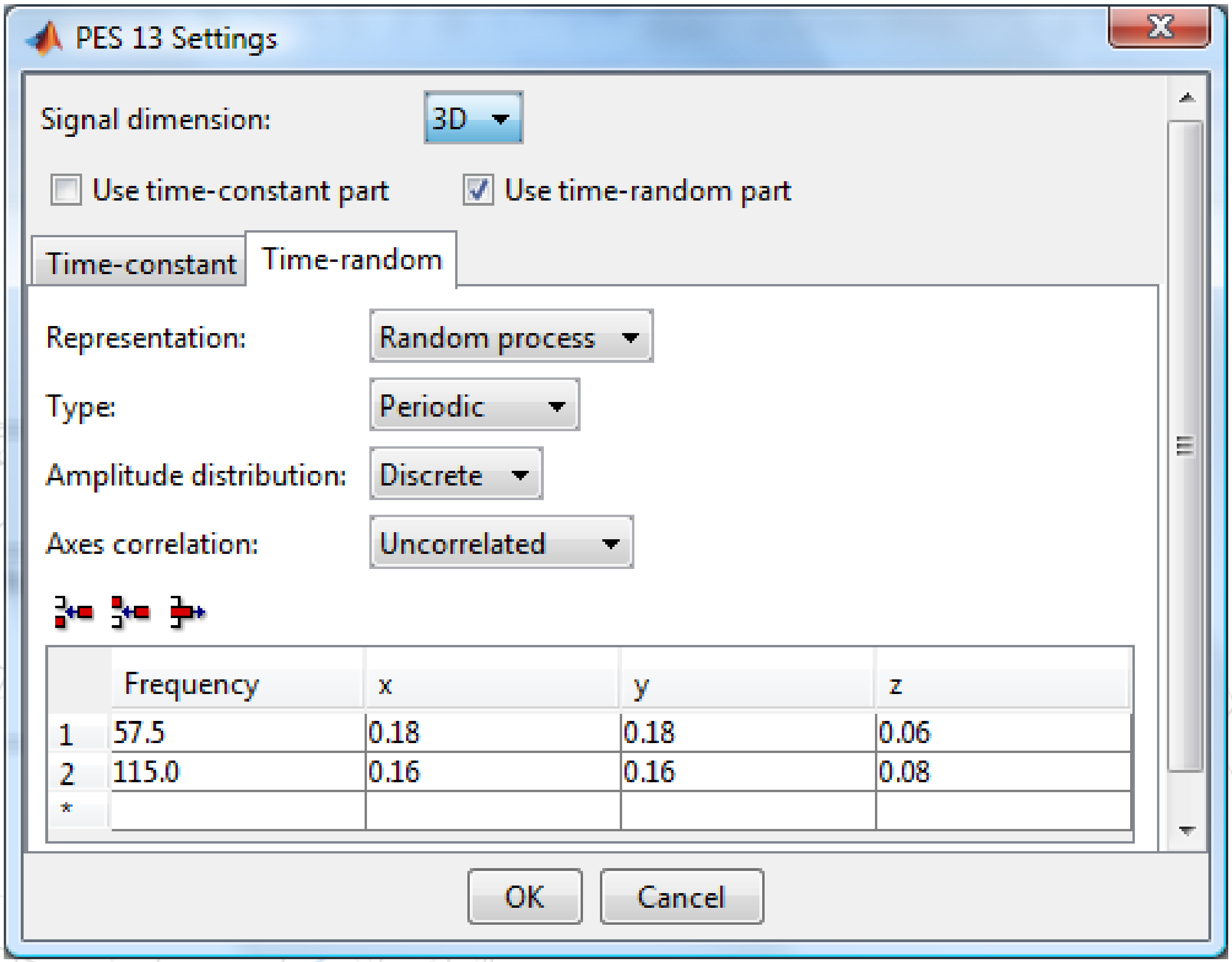
Example: Cryocooler micro-vibrations

- Measured harmonics of compressor forces at a set of frequencies
- ➡ Random process of type Periodic
- ➡ Definition via distinct force-amplitude pairs in PEET



$$\boldsymbol{\alpha}_{14} = \begin{bmatrix} P(\mathbf{f}_{14}, \mathbf{A}_{14,x}) \\ P(\mathbf{f}_{14}, \mathbf{A}_{14,y}) \\ P(\mathbf{f}_{14}, \mathbf{A}_{14,z}) \end{bmatrix} \text{mN} = \begin{bmatrix} P([57.5, 115]\text{Hz}, [180, 160]) \\ P([57.5, 115]\text{Hz}, [180, 160]) \\ P([57.5, 115]\text{Hz}, [60, 80]) \end{bmatrix} \text{mN}$$

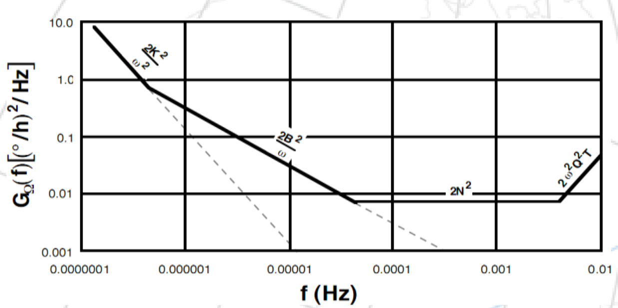
Periodic Random Process PES

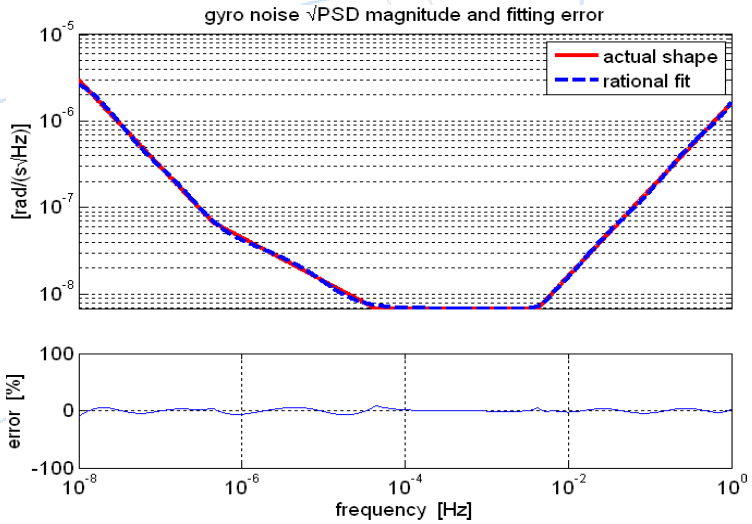


Gyro Rate Noise PES (AST-1)

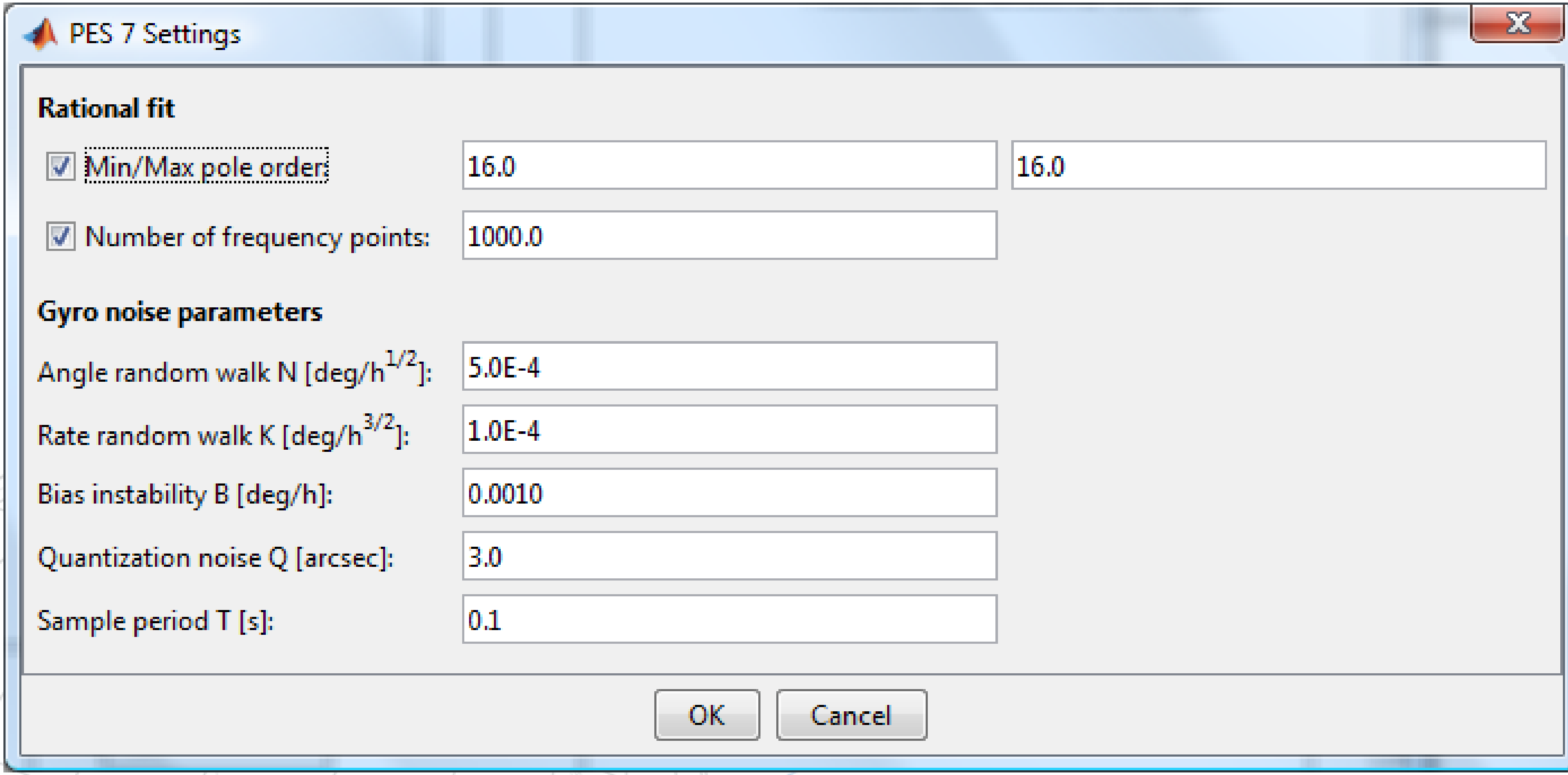
- Given by a set of typical parameters describing piece-wise the spectrum
  - ➡ Random process described as PSD
  - ➡ Special input dialog

Angle random walk: 0.0005°/√h
Bias instability: 0.005°/h
Rate random walk: 0.0001°/$h^{3/2}$
Quantization noise: 3 arcsec
Sample period: 0.1 sec





Gyro Noise Block Dialog



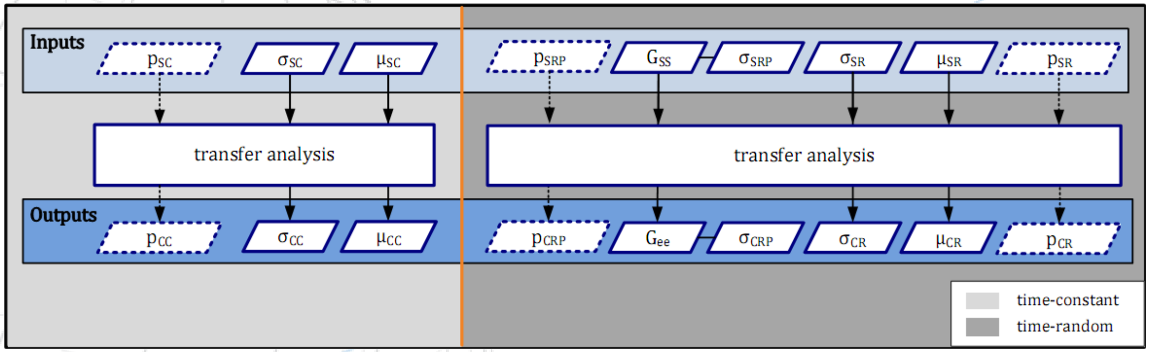
AST-2

Transfer Analysis

Transfer Analysis (AST-2)

Transfer Analysis:

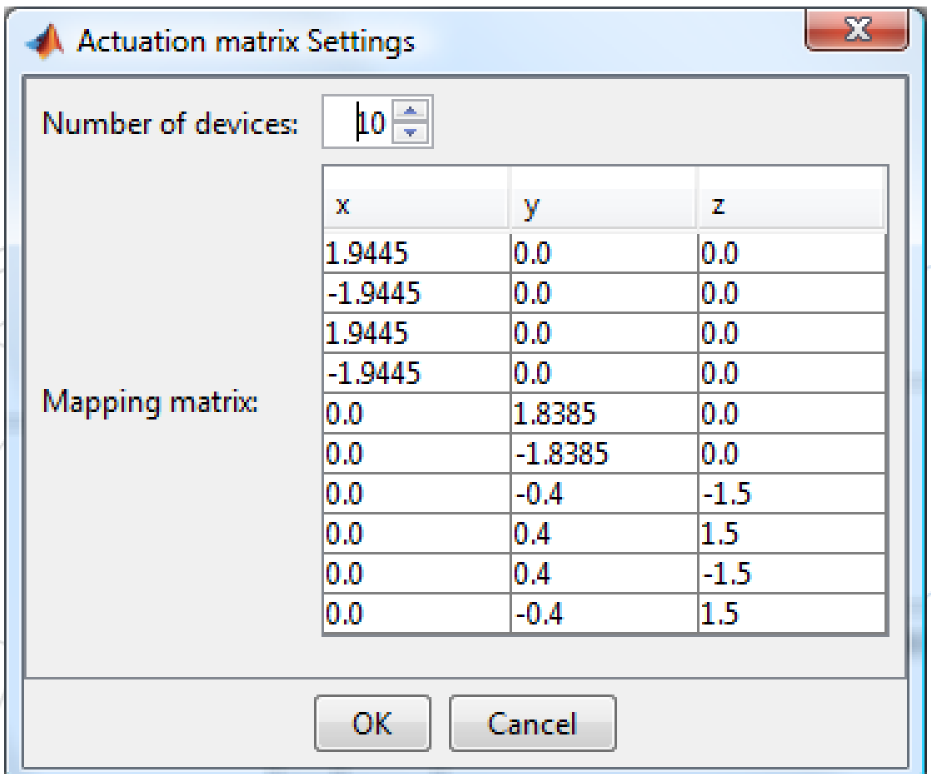
- Transfer from PES to contributors on pointing level via
  - static systems (e.g. coordinate transformations, actuation matrix,…)
  - dynamic systems (e.g. plant, controller, estimator, structure)
- System transfer definition remains user task
  - but PEET provides tools for realization and basic models



Mapping Matrix

Example: Thruster noise

- Mapping of ('1D') thrust noise to physical axes via thruster configuration matrix



x	y	z
1.9445	0.0	0.0
-1.9445	0.0	0.0
1.9445	0.0	0.0
-1.9445	0.0	0.0
0.0	1.8385	0.0
0.0	-1.8385	0.0
0.0	-0.4	-1.5
0.0	0.4	1.5
0.0	0.4	-1.5
0.0	-0.4	1.5

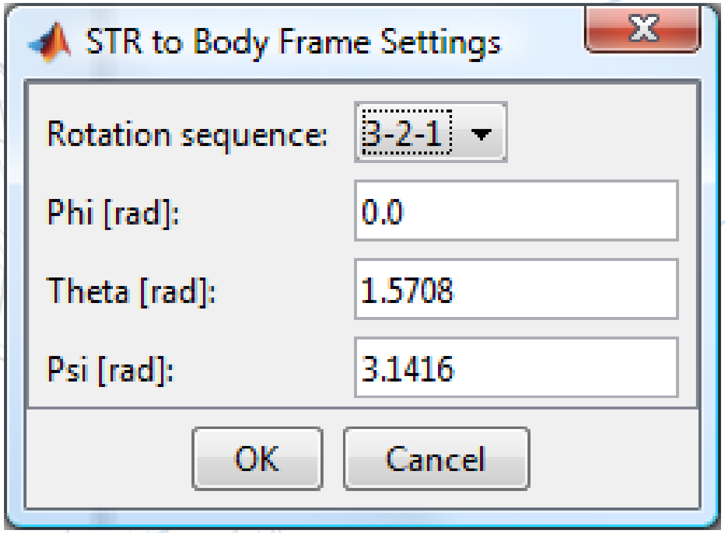
Number of 'devices' n

n x 3 mapping data

Coordinate Transformation

Example: Star tracker temporal noise

- Conversion from sensor to pointing axes via coordinate transformation
- Defined by rotation sequence and three angles

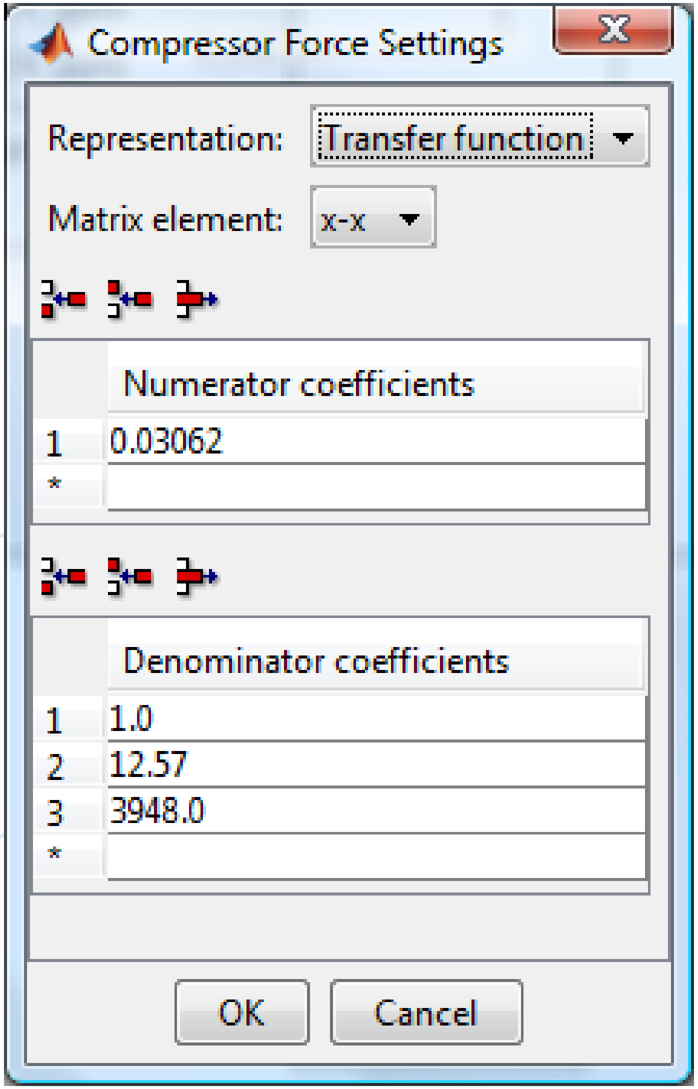


Dynamic System Block (AST-2)

Example: Cryocooler micro-vibrations

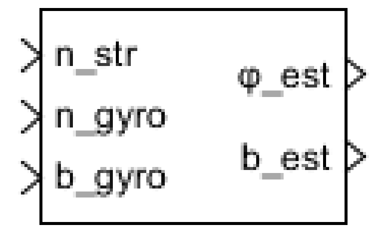
- Transfer of compressor forces to pointing error level via satellite structure

- Alternative definitions:
  - Zero-Pole-Gain model
  - State-Space model



Gyro Stellar Estimator (AST-2)

- SISO fixed-gain Kalman-Filter for each axis
- Possible inputs:
  - Star Tracker attitude
  - Gyro noise
  - Gyro bias
- Possible outputs:
  - Attitude and drift bias estimation error
- Parameters:
  - 2 Kalman gains for each axis

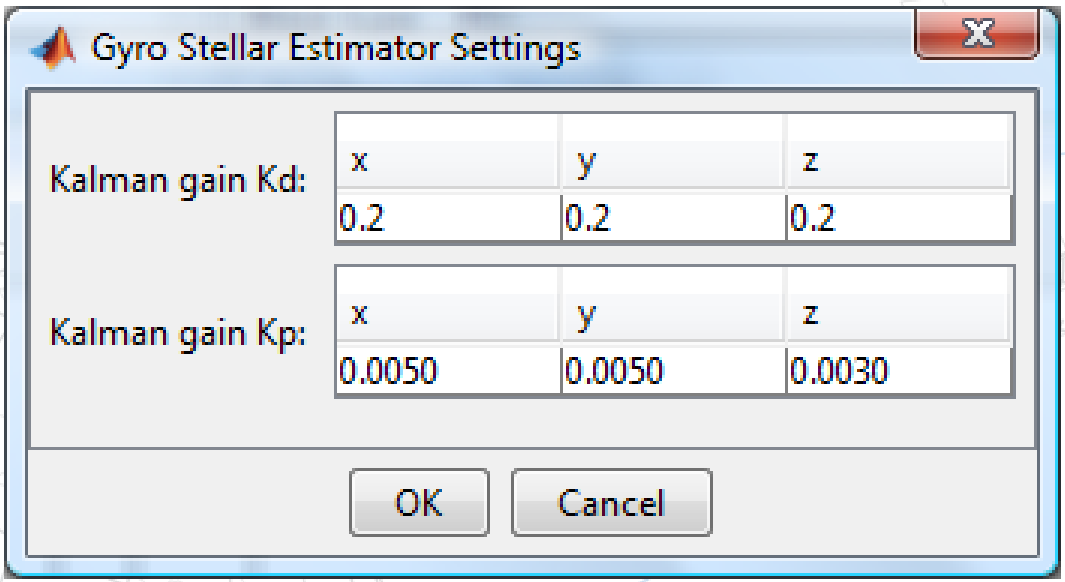


Gyro Stellar Estimator

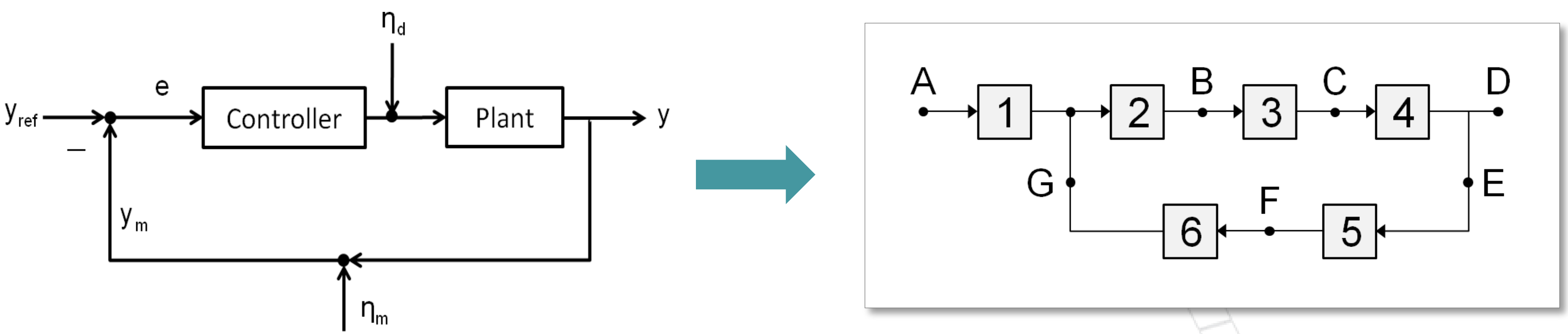
$$\mathbf{K}_1 = \begin{bmatrix} K_{1,x} \\ K_{1,y} \\ K_{1,z} \end{bmatrix} = \begin{bmatrix} 0.2 \\ 0.2 \\ 0.2 \end{bmatrix}$$

$$\mathbf{K}_2 = \begin{bmatrix} K_{2,x} \\ K_{2,y} \\ K_{2,z} \end{bmatrix} = \begin{bmatrix} 0.005 \\ 0.005 \\ 0.003 \end{bmatrix}$$

Gyro Stellar Estimator (AST-2)

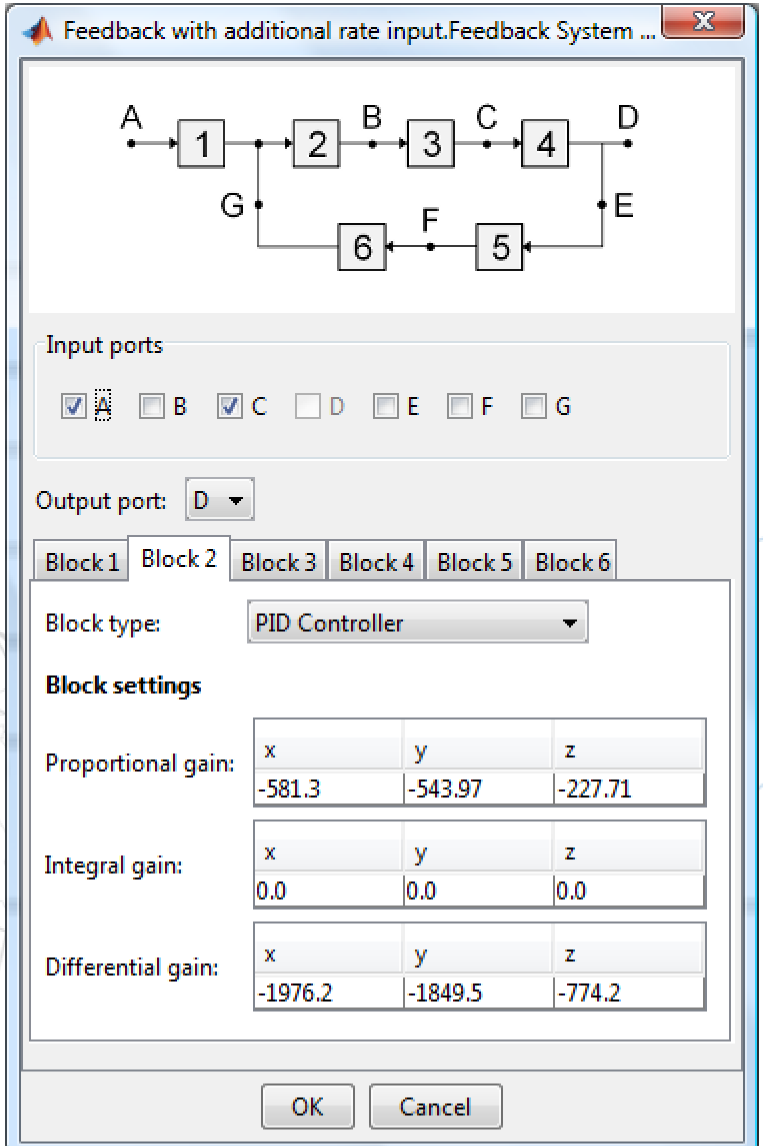


Feedback Block (AST-2)

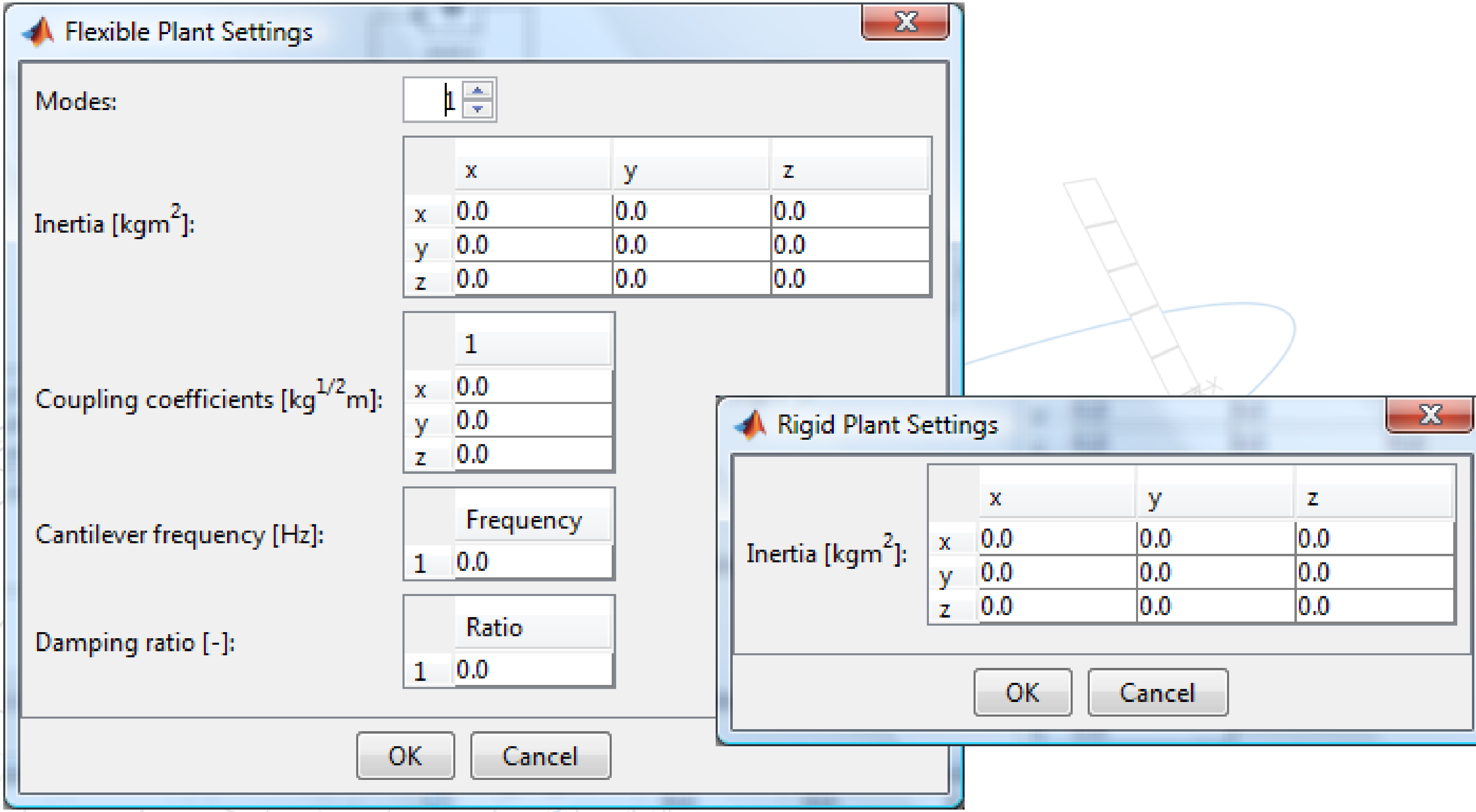


- Fixed internal block structure, but
- Each internal block is customizable
  - User can select from any SISO transfer block defined in the library
  - E.g. dynamic system, plant, controller, etc.
- User-defined input nodes and user-defined output node
- Remaining sub-blocks not required (unity transfer by default)

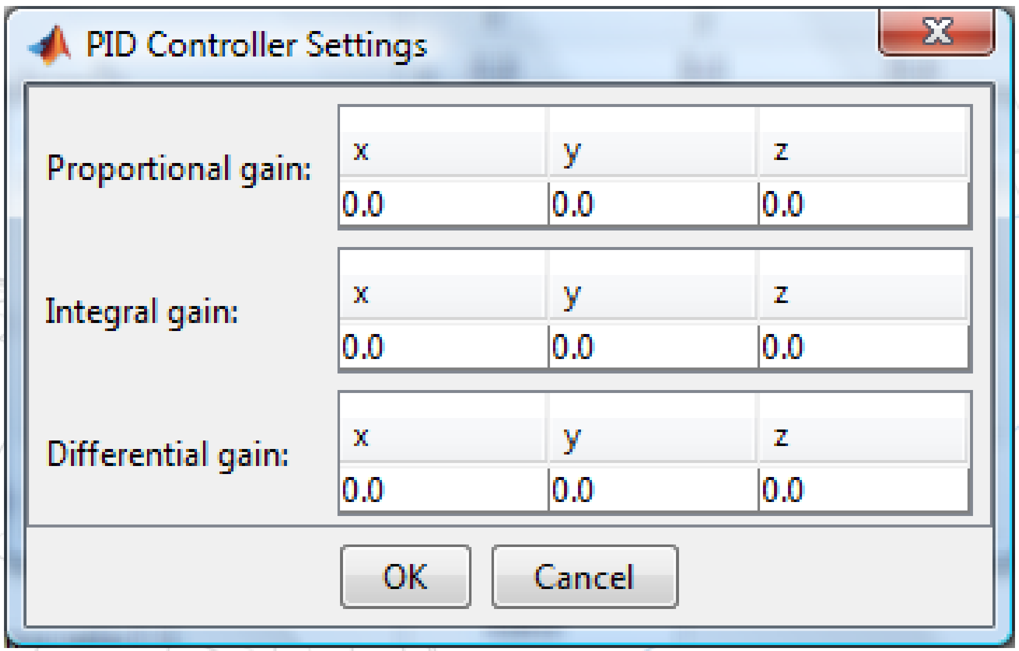
Feedback Block (AST-2)



Flexible and Rigid Plant



PID controller



AST-3 & AST-4

Pointing Error Index Contribution
&
Pointing Error Evaluation

PE Index Contribution (AST-3)

Working steps of AST-3:

1. Worst case pointing error index
   → user

2. Application of pointing error metrics
   → PEET

3. Statistical interpretation of pointing error indices
   → PEET

Pointing Error Evaluation (AST-4)

Working steps of AST-4:

1. Define correlation between PEC

   ➡ user & PEET

2. Summation of PEC and level of confidence evaluation

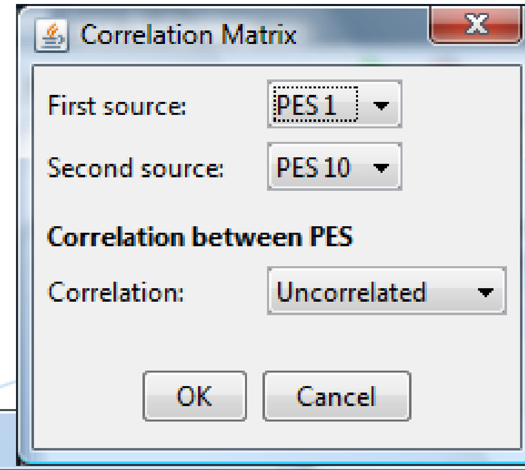
   ➡ PEET

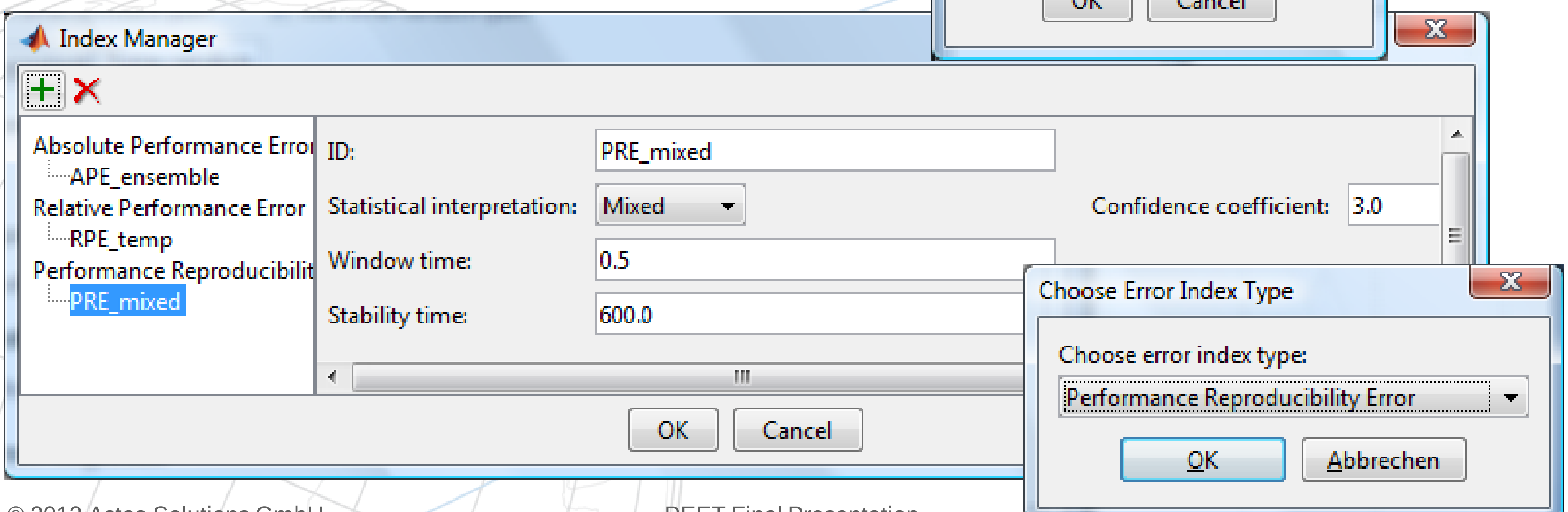
3. Compilation of total pointing error

   ➡ PEET

PE Index Contribution & Evaluation

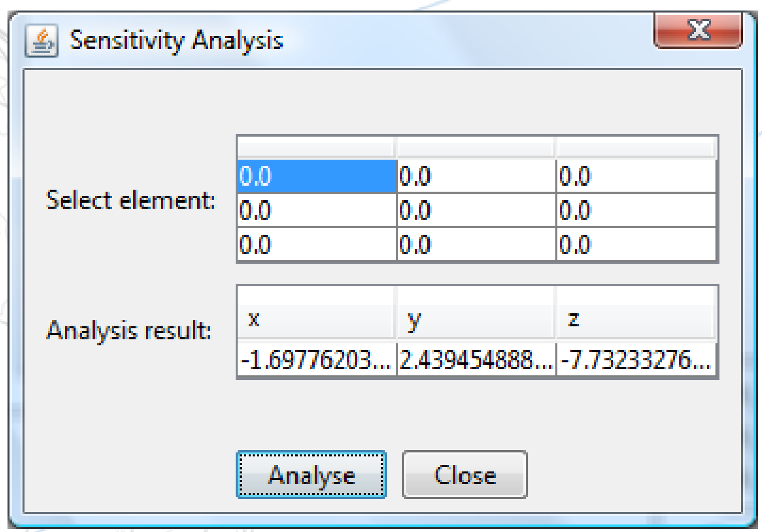
- Processing of AST-3 and AST-4 according to global user settings
  - selection of pointing error index
  - level of confidence
  - window time & stability time
  - statistical interpretation
  - correlation





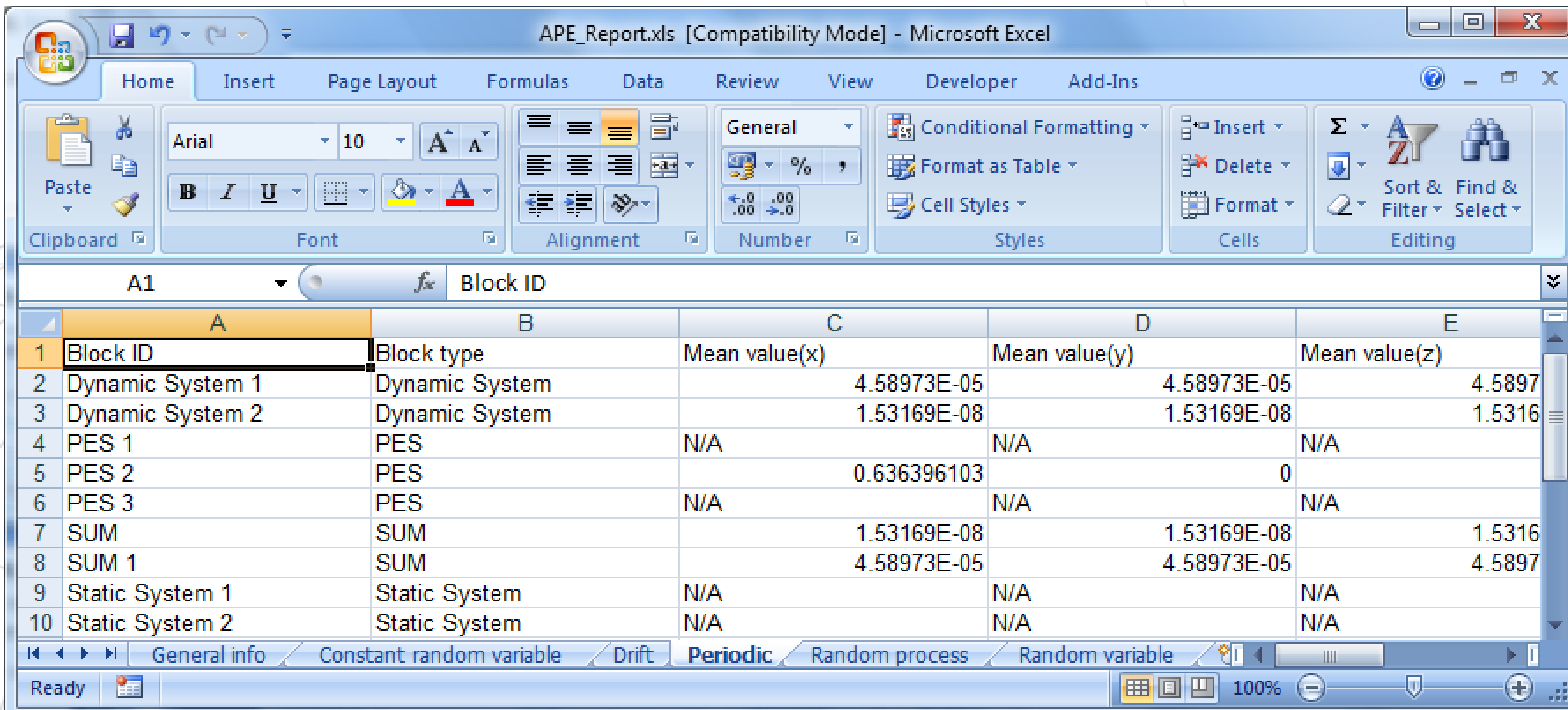
Pointing Error Break-Down

- Pointing Error Break-Down is supported by PEET in terms of a sensitivity analysis (identification of error drivers)
- For nearly all input parameters it is possible to analyse the total pointing error sensitivity with respect to these parameters
- PEET displays the sensitivity as differential quotient for each axis.



Result Export

- Data structures accessible from MATLAB (workspace)
- Export via Excel interface (single block data)
- Reporting function (one Excel document for each index)



	A	B	C	D	E
1	Block ID	Block type	Mean value(x)	Mean value(y)	Mean value(z)
2	Dynamic System 1	Dynamic System	4.58973E-05	4.58973E-05	4.5897
3	Dynamic System 2	Dynamic System	1.53169E-08	1.53169E-08	1.5316
4	PES 1	PES	N/A	N/A	N/A
5	PES 2	PES	0.636396103	0	
6	PES 3	PES	N/A	N/A	N/A
7	SUM	SUM	1.53169E-08	1.53169E-08	1.5316
8	SUM 1	SUM	4.58973E-05	4.58973E-05	4.5897
9	Static System 1	Static System	N/A	N/A	N/A
10	Static System 2	Static System	N/A	N/A	N/A

Expandability of PEET

The block library of PEET can be extended in several ways:

- Writing own MATLAB classes (interfaces are explained in the ICD)
- Adding block containers that comprise a multi-block subsystem
- Adding basic blocks with predefined parameters (masked blocks)

PEET Website

A website for PEET will be available soon.

- Overview
- Documentation
- Downloads
- Bug tracker and wish list
- Sections on pointing error engineering standard and handbook

http://peet.estec.esa.int

Key Features / Summary

- Workflow in Accordance with PEEH
- Automatically performs step AST-3 and AST-4 of the PEEH
- Fast switching between Pointing Error Indices
- Extendable block library (low, medium and high level models)
- Flexible definition of PES (nearly no conversions have to be made in advance by the user)
- Built-in plotting of PSDs and time series
- Performs error budgeting and supports break-down analyses
- Fully integrated in MATLAB environment (input parameters may be defined as MATLAB workspace variables)
- Available to industry (of ESA member countries) for free
  - Standardized exchange format
  - Easy model exchange between contractor(s) and ESA

Current Status of PEET Prototype

- Prototype development finalized
- All presented functionality is available
- User and developer documentation available
- Verification and basic validation is made (using the PointingSat example and other basic testcases)
- A detailed validation of all potential block combination would be desireable but a lot of effort (> 1000 testcases; not part of the project)

Outlook

- Extensive validation campaign
- Extension and refinement of the core routines, e.g.
  - Optimization of prototype routines (performance improvement)
  - Extended correlation options (currently only full or none)
- Extension of user-interface, e.g.
  - Extension of block database with more parametric model for subsystems
  - Advanced PES and dynamic system definition
- Extension of analysis capability
- Tutorials and handbook improvements (according to leassons learnt)