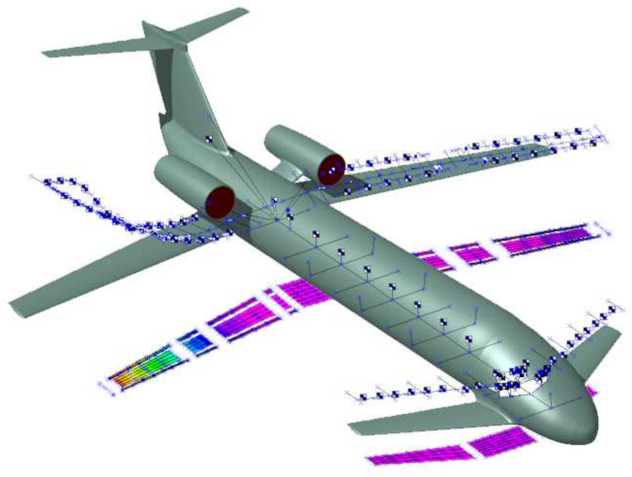
NeoCASS

Next generation Conceptual Aero Structural Sizing



Dipartimento di Scienze e Tecnologie Aerospaziali
Politecnico di Milano

Report name	NeoCASS Next generation Conceptual Aero Structural Sizing
Date	December 11, 2013
Partner Politecnico di Milano	Dipartimento di Scienze e Tecnologie Aerospaziali
Author	L. Cavagna, S. Ricci
Delivery number	D.XXXX
Dissemination level	Public

Contents

List of Figures

List of Tables

Appendix A

NeoRESP-SS: state-space aeroelastic response

NeoRESP-SS allows to solve for the dynamic reponse of the aircraft to external excitations, i.e. gusts, controls, generalized forces, in the time domain. While NeoRESP directly solves for dynamic equation in the frequency domain, NeoRESP-SS defines a state-space model for aerodynamics. This model is then coupled to the mechanical one and the response is recovered through time-domain integration.
The input file data for the two solvers is the same. Once the pre-processor has been run, the user can choose which one to run.

A.1 Solver formulation

The dynamic equations to be solved are:

$$\mathbf{M}\ddot{q}(t) + \mathbf{C}\dot{q}(t) + \mathbf{K}q(t) = q_\infty Q_{am}(t) + q_\infty Q_{ag}(t) + q_\infty Q_{a\delta}(t) + Q_q(t) \tag{A.1}$$

with:

- $\mathbf{M}$, $\mathbf{C}$, $\mathbf{K}$ respectively the mass, damping and stiffness matrix of the mechanical system;
- $q(t)$ the modal coordinates;
- q_∞ the dynamic reference pressure, $q_\infty = \frac{1}{2}\rho V_\infty^2$;
- Q_{am}, Q_{ag}, $Q_{a\delta}$ respectively the aerodynamic forces due to structural motion, gusts and controls;
- Q_q generalized external forces, $Q_q = \mathbf{U}_h^T F_0 f(t)$ where $\mathbf{U}_h$ is a subset of modal shapes at the loaded nodes, F_0 is the load shape and $f(t)$ its time-varying amplitude.

NeoRESP-SS creates a state-space model for Q_{am}, Q_{ag}, $Q_{a\delta}$ using the method presented in [1].
Instead of creating one single model for aerodynamics as it should be in reality, the current release adopts three independent state-space models for Q_{am}, Q_{ag}, $Q_{a\delta}$ in order to:

- allow future enhancements to continuos turbulence with the introduction of gust shape filters;
- allow the user to have hands-on each excitation force and model it with the most suitable number of states without affecting the others;
- change independently the approximation of each model, keeping the others unchanged.

Considering Q_{am}, the following state-space system can be determined:

$$\begin{aligned} \dot{x}_{am}(t) &= \frac{V_\infty}{l_a}\mathbf{A}_{am}x_{am}(t) + \frac{V_\infty}{l_a}\mathbf{B}_{0am}q(t) + \mathbf{B}_{1am}\dot{q}(t) + \frac{l_a}{V_\infty}\mathbf{B}_{2am}\ddot{q}(t) \\ Q_{am}(t) &= \mathbf{C}_{am}x_{am}(t) + \mathbf{E}_{0am}q(t) + \frac{l_a}{V_\infty}\mathbf{E}_{1am}\dot{q}(t) + \left(\frac{l_a}{V_\infty}\right)^2\mathbf{E}_{2am}\ddot{q}(t) \end{aligned} \tag{A.2}$$

where x_{am} are the aerodynamic states, q the structural modes, V_∞ the reference velocity and l_a the reference length. The same can be written for the case of gust inputs with angle profile $\alpha_g = \frac{v_g(t)}{V\infty}$:

$$\begin{aligned} \dot{x}_{ag}(t) &= \frac{V_\infty}{l_a}\mathbf{A}_{ag}x_{ag}(t) + \frac{V_\infty}{l_a}\mathbf{B}_{0ag}\alpha_g(t) + \mathbf{B}_{1ag}\dot{\alpha}_g(t) + \frac{l_a}{V_\infty}\mathbf{B}_{2ag}\ddot{\alpha}_g(t) \\ Q_{ag}(t) &= \mathbf{C}_{ag}x_{ag}(t) + \mathbf{E}_{0ag}\alpha_g(t) + \frac{l_a}{V_\infty}\mathbf{E}_{1ag}\dot{\alpha}_g(t) + \left(\frac{l_a}{V_\infty}\right)^2\mathbf{E}_{2ag}\ddot{\alpha}_g(t) \end{aligned} \tag{A.3}$$

or control deflections $\delta(t)$:

$$\begin{aligned} \dot{x}_{a\delta}(t) &= \frac{V_\infty}{l_a}\mathbf{A}_{a\delta}x_{a\delta}(t) + \frac{V_\infty}{l_a}\mathbf{B}_{0a\delta}\delta(t) + \mathbf{B}_{1a\delta}\dot{\delta}(t) + \frac{l_a}{V_\infty}\mathbf{B}_{2a\delta}\ddot{\delta}(t) \\ Q_{a\delta}(t) &= \mathbf{C}_{a\delta}x_{a\delta}(t) + \mathbf{E}_{0a\delta}\delta(t) + \frac{l_a}{V_\infty}\mathbf{E}_{1a\delta}\dot{\delta}(t) + \left(\frac{l_a}{V_\infty}\right)^2\mathbf{E}_{2a\delta}\ddot{\delta}(t) \end{aligned} \tag{A.4}$$

The equations given in Eq. (A.1) are written in a descriptor form, as:

$$\mathbf{E}_{ae}\dot{x}_{ae}(t) = \mathbf{A}_{ae}\dot{x}_{ae}(t) + \mathbf{B}_{ae}u(t) \tag{A.5}$$

with:

$$\mathbf{E}_{ae} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M} - q_\infty\mathbf{E}_{2am} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & -\frac{l_a}{V_\infty}\mathbf{B}_{2am} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \tag{A.6}$$

$$\mathbf{A}_{ae} = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{K} - q_\infty\mathbf{E}_{0am} & \mathbf{C} - q_\infty\mathbf{E}_{1am} & q_\infty\mathbf{C}_{am} & q_\infty\mathbf{C}_{ag} & q_\infty\mathbf{C}_{a\delta} \\ \frac{V_\infty}{l_a}\mathbf{B}_{0am} & \mathbf{B}_{1am} & \frac{V_\infty}{l_a}\mathbf{A}_{am} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{V_\infty}{l_a}\mathbf{A}_{ag} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{V_\infty}{l_a}\mathbf{A}_{a\delta} \end{bmatrix} \tag{A.7}$$

$$
\mathbf{B}_{ae} = \begin{bmatrix}
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
q_\infty \mathbf{E}_{0ag} & q_\infty \mathbf{E}_{1ag} & q_\infty \mathbf{E}_{2ag} & q_\infty \mathbf{E}_{0a\delta} & q_\infty \mathbf{E}_{1a\delta} & q_\infty \mathbf{E}_{2a\delta} & \mathbf{U}_q^T F_0 \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\frac{V_\infty}{l_a}\mathbf{B}_{0ag} & \mathbf{B}_{1ag} & \frac{l_a}{V_\infty}\mathbf{B}_{2ag} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{V_\infty}{l_a}\mathbf{B}_{0a\delta} & \mathbf{B}_{1a\delta} & \frac{l_a}{V_\infty}\mathbf{B}_{2a\delta} & \mathbf{0}
\end{bmatrix} \tag{A.8}
$$

$$
\begin{aligned}
x_{ae}^T &= \{q^T, \dot{q}^T, x_{am}^T, x_{ag}^T, x_{a\delta}^T\} \\
u^T &= \left\{\alpha_g^T, \dot{\alpha}_g^T, \ddot{\alpha}_g^T, \delta^T, \dot{\delta}^T, \ddot{\delta}^T, f^T\right\}
\end{aligned} \tag{A.9}
$$

Finally, outputs are available as:

$$
Y_{ae} = \mathbf{C}_{ae}\dot{x}_{ae}(t) + \mathbf{D}_{ae}u(t) \tag{A.10}
$$

where Y_{ae} is by default:

$$
Y_{ae}^T = \left\{x_{ae}^T, \ddot{q}^T, Cx_{am}^T, Cx_{ag}^T, Cx_{a\delta}^T, H_\delta^T\right\} \tag{A.11}
$$

with the aerodynamic coefficients Cx_{ai}:

$$
Cx_{ai}^T = \{Cy_{ai}, Cz_{ai}, Cl_{ai}, Cm_{ai}, Cn_{ai}\}^T \quad i = m,\, g,\, \delta \tag{A.12}
$$

and the hinge moments H_δ.
Nodal displacements and element forces can be recovered through `DISP` and `IFORCE` cards.

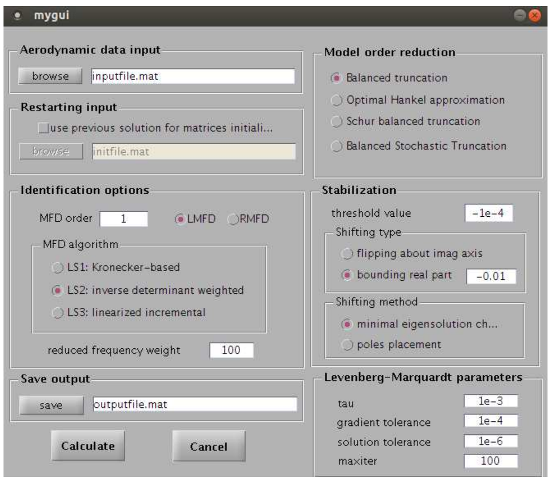
A.2 Running the solver

Once the preprocessor has been run, the database with all the data required is available and can be saved for future runs.
At the first run, the user calls for the solver:
`solve_free_lin_trim_ss()`
without provinding any input. According to the model source of excitations, from one to three `.mat` files are exported, one for Q_{am}, Q_{ag}, and $Q_{a\delta}$ respectively. For example, in the case of gust input only, the following message appears:

```
- Aerodynamic matrix Ham exported to model_Ham_M_0.7.mat file for fitting.
   Rows: 12.
   Columns: 12.
   Extra outputs: 5.

 - Aerodynamic matrix Hag exported to model_Hag_M_0.7.mat file for fitting.
   Rows: 12.
   Columns: 1.
   Extra outputs: 5.
```

where `model` is the main filename with the model. The two binary files available are ready to be processed to define all the state-space matrices required in Eq. (A.5).


mygui

Aerodynamic data input
browse | inputfile.mat

Restarting input
[ ] use previous solution for matrices initiali...
browse | initfile.mat

Identification options
MFD order: 1 — [✓ LMFD] RMFD
MFD algorithm:
- LS1: Kronecker-based
- [✓ LS2: inverse determinant weighted]
- LS3: linearized incremental

reduced frequency weight: 100

Save output
save | outputfile.mat

Model order reduction
- [✓ Balanced truncation]
- Optimal Hankel approximation
- Schur balanced truncation
- Balanced Stochastic Truncation

Stabilization
threshold value: -1e-4
Shifting type:
- flipping about imag axis
- [✓ bounding real part] -0.01

Shifting method:
- [✓ minimal eigensolution ch...]
- poles placement

Levenberg-Marquardt parameters

tau	1e-3
gradient tolerance	1e-4
solution tolerance	1e-6
maxiter	100

Figure A.1: GUI for the fitting process.

A.2.1 Fitting

The fitting process for the generation of the aerodynamic state-space modelss can be carried out in two ways:

- running `mygui` at the prompt, filling-in the GUI panel showed in Fig. A.2.1 and calculating the fitting until results are satisfactory;
- make a local copy of the script `aero_ss`, open the file, modify the parameters and run the script until results are satisfactory.

In the first case, the user has to:

1. load the `.mat` file exported by NeoRESP (Aerodynamic data input)
2. select the order of the approximation (Identification options - MFD order) and between Left/Right Matrix Fraction Description (LMFD or RMFD)
3. select the algorithm (LS1, LS2, LS3). LS1 should be used only for small cases because of its computational cost.
4. select the weight for the second value of reduced frequency. This parameter is used to enforce zero frequency constraints for generalized forces and its derivatives.
5. pressing the button `Calculate` will start the calculation and export a file with the state-space model with the name given in Save Output panel. At the end of the process, an interactive menu at the prompt will allow to display the quality of the fitting.

If the approximation is poor, the user can repeat the steps from 2 to 5, typically raising the order of the approximation. Usually the order ranges from 2 to 4. However, if the order is too large, the aerodynamic model may introduce not-observable and non-controllable dynamics to the system. To avoid this, the model can be reduced during the fitting process through the following methods (Model Order Reduction):

- balanced model truncation via square root method (BALANCED);
- Hankel minimum degree approximation (MDA) without balancing (HANKEL);
- balanced model truncation via Schur method (SCHUR);
- balanced stochastic model truncation (BST).

Hankel Singular Values are plotted on the screen and the user can decide, on the basis of their magnitude, how many of them will be retained.
The stabilization panel allows the user to enforce the stability of the matrices $\mathbf{A}_{am}$, $\mathbf{A}_{ag}$, $\mathbf{A}_{a\delta}$. Enforcement of fitting stability is achieved using minimal eigensolution changes or through a poles placement technique. Both methods check the eigensolutions stability, threshold value, at each iteration of the identification. If any eigenvalue is below an assigned stability threshold value it is shifted to a user given admissible real negative eigenvalue (bounding real part). Then either the related eigenvectors are modified so to mantain the original structure, eigshift, of the state matrix or a pole placement, a polesplace, of the newly stabilized eigenspectrum is applied. The choice of the threshold value depends on the model being considered. After the fitting process has been carried out, eigenvalues of these matrices are reported on the screen in terms of reduced frequency. Too small a value may mean an unrealistic slow representation of aerodynamic transients, i.e. the number of aerodynamic chords the transient completes. Finally, the following parameters ruling the Levenberg-Marquardt algorithm can be modified:

- `tau`, defining the starting value for the Levenberg-Marquardt damping parameter (lambda) as lambda = tau*max(Jacobian'*Jacobian);
- `gradient tolerance` on the solution gradient, below which the algorithm stops;
- `solution tolerance` on the solution variation, below which the algorithm stops;
- `maxiter` maximum number of iterations.

These parameters need not to be changed.
The same process can be run through the script `aero_ss` available in `NeoRESP/State_space_dyn` folder. The user can modify the following lines:

```
%
% Approximation
%
opt{1} = 4;      % MFD order
opt{2} = 3;      % MFD algorithm
opt{3} = 'lmfd'; % left or right MFD
opt{4} = 2;      % residualization order
opt{5} = 100;    % weight parameter value W^2
%
% Levenberg-Marquardt parameters
%
tau     = 1e-3;
tolg    = 1e-4;
```

```
tolx    = 1e-6;
maxiter = 100;
%
% Stabilization
%
eigsopt.threshold = -1e-3;
eigsopt.type      = 'bound';      % 'bound' or 'flip'
eigsopt.bound     =  -1.5e-1;
eigsopt.method    = 'polesplace'; %'polesplace' or 'eigshift'
%
% Model reduction algorithm
%
algROM = 'bst'; % 'schur', 'hankel', 'balance'
```

and run the script as:
`aero_ss(input_mat_file, output_mat_file)`, where `input_mat_file` is the file created by NeoRESP and `output_mat_file` is the output file with the state space model. For example, `aero_ss('model_Ham_M_0.7.mat', 'res1.mat')` will load the file `model_Ham_M_0.7.mat`, run the fitting and export the results in `res1.mat` file.
The adoption of the script `aero_ss` enables to keep track of the parameters used and is hence the most preferable of the two.

A.2.2 Model assembly and solution

Once the fitting has been carried out, the user runs again the solver:

```
[ss_model,Y,T,X,U] =
    solve_free_lin_dyn_ss('Tmax',15,'dT',1e-3,'Ham','res1.mat','Hag','res2.mat');
```

providing the following inputs:

- `Tmax` (not mandatory), the maximum simulation time;
- `dT` (not mandatory), simulation time step;
- `Ham` (mandatory), output filename from the fitting process for Q_{am};
- `Hag` (mandatory if the gust input is present), output filename from the fitting process for Q_{ag};
- `Had` (mandatory if the control input is present), output filename from the fitting process for $Q_{a\delta}$;

The solver runs and gives the following outputs:

- `ss_model`, the state space model given in Eq. (A.1) in canonical form;
- `Y`, is the output Y_{ae} provided by Eq. (A.10);
- `T`, a vector with simulation time, `T = [0:dT:Tmax]`;

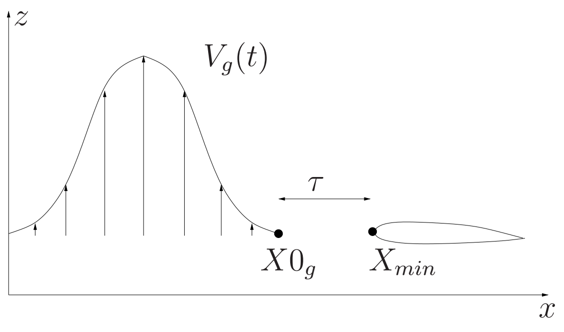


Figure A.2: Gust delay τ applied to model input u.

- `X`, the state x_{ae} ;
- `U`, the input vector u;

The state space model `ss_model` can be visualized, modified at the command prompt or loaded into Simulink. By typing at the prompt the name of the model, i.e.`ss_model` in this case, the main features of the state system are displayed:

```
Input groups:
         Name          Channels
      gust_alpha          1
    gust_alphadot         2
    gust_alphaddot        3

Output groups:
       Name                                  Channels
        q                               1,2,3,4,5,6,7,8,9,10,11,12
      qdot                          13,14,15,16,17,18,19,20,21,22,23,24
      qddot                         25,26,27,28,29,30,31,32,33,34,35,36
   node_acc                                  37,38,39,40,41,42
  bar_forces 43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66
aero_force_mode                         67,68,69,70,71,72,73,74,75,76,77,78
aero_force_gust                         79,80,81,82,83,84,85,86,87,88,89,90
aero_coeff_mode                                   91,92,93,94,95
aero_coeff_gust                                   96,97,98,99,100
```

together with state-space matrices. The channels identify the range of the index each group of variable has in the array u or Y_{ae}. For example, modal amplitudes q can be recovered from the first twelve elements in Y_{ae}; aerodynamic coefficients due to motion, i.e. C_y, C_z, C_l, C_m, C_n, are labelled as `aero_coeff_mode` and are stored in Y_{ae} from index 91 up to 95.
Internal delays are introduced to model inputs u. For control surfaces or external forces, delays are directly taken from the field available respectively in `SURFDEF` and `DLOAD` cards. For gusts, the delay is treated as follow. A local reference point X_{min} is placed at the first panel along freestream (see Fig. A.2.2). The transfer matrix for gust generalized forces

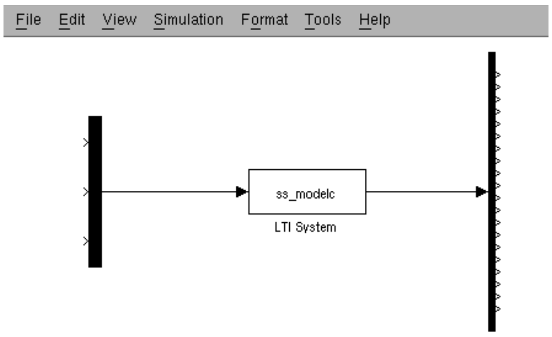


Figure A.3: State space model in Simulink.

$\mathbf{H}_{ag}(jk, M_\infty)$ is hence evaluated in this reference system. The purpose of such a choice is to try to limit the spiraling behavior for $\mathbf{H}_{ag}$. The smoother its terms, the easier the fitting process. A delay τ is then applied to the inputs α_g, $\dot{\alpha}_g$, $\ddot{\alpha}_g$, as:

$$\tau = \frac{X_{min} - X0_g}{V_\infty} \tag{A.13}$$

where $X0_g$ is the user defined gust entry point provided in the `GUST` card. Model delays can be found in `ss_model.InputDelay`.
The user can run the model outside NeoRESP by typing:
`[Y,T,X] = lsim(ss_model, U', T);`
This can be helpful for a gust assessment for example. The user can redefine the vector u according to a new gust shape and launch the simulation.
Finally, the state model can be imported in Simulink using LTI system block. Mux and Demux blocks are used to include respectivelty inputs and outputs, as showed in Fig. A.2.2.
While NeoRESP runs, roots for different models are reported:

1. structural model, i.e. $(-\omega^2\mathbf{M} + \mathbf{K})\ q = 0$
2. aerodynamic model only for Q_{am}, i.e. $\left(\lambda\mathbf{I} - \frac{V_\infty}{l_a}\mathbf{A}_{am}\right)\ x_{am} = 0$
3. aeroelastic system given by the coupling of structure and aerodynamic system for Q_{am};
4. complete system given by the coupling of structure, aerodynamic system for Q_{am}, Q_{ad} and Q_{ag}, i.e.$(\lambda\mathbf{E}_{ae} - \mathbf{A}_{ae})\ x_{ae} = 0$

This allows to see how roots change during the assembly of the final model and helps the user to discover where instabilities at point 3 and 4 may come from (if any).

Bibliography

[1] M. Ripepi and P. Mantegazza, *Improved matrix fraction aproximation of aerodynamic transfer matrices*, AIAA Journal **51** (2013), no. 5, 1156–1173.