pyBADA

Easy Base of Aircraft Data (BADA) integration in Python for rapid prototyping

UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

Introduction

- **BADA** is an aircraft performance model (**APM**) developed and maintained by **EUROCONTROL.**
- **pyBADA** is a **Python** library for aircraft performance **modelling**, trajectory **prediction** and **optimisation**, and **visualisation** with BADA, build on **CasADi**, an open-source tool for optimisation and algorithmic differentiation.
- pyBADA is **multi-platform** and easy to install → **pip install ssh://user.name@bleriot.upc.edu/path/pyBada**
- pyBADA implements:
  - BADA3, BADA4
  - BADAH

CasADi

- At the core of CasADi is a **symbolic framework** that allows the user to construct symbolic expressions.

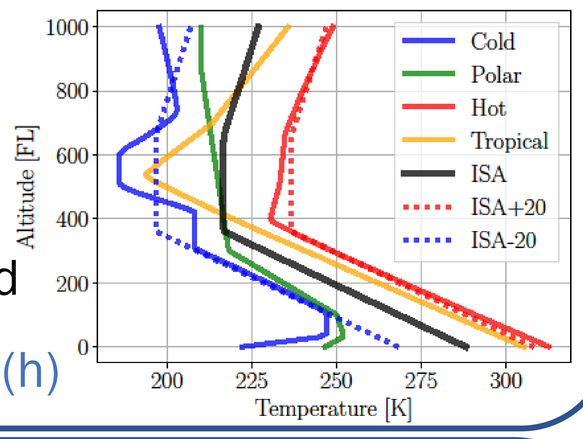
Numeric:
```
x = 3
y = 5
z = x^2 + y
>14
```

Symbolic:
```
x = SX.sym('x')
y = SX.sym('y')
z = x^2 + y
>x^2 + y
gradient(z,x) >2*x
```

Atmophere

- Implements **the standard atmosphere models:**
  - ISA (international standard atmosphere)
  - MIL-STD-210A standard

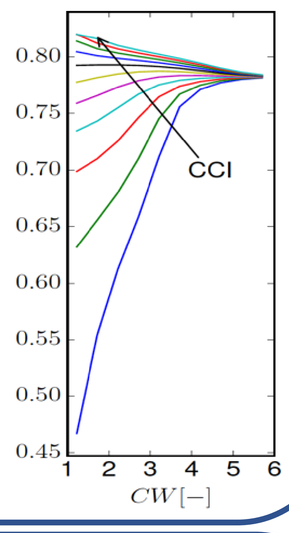
```
theta = atmosphere.theta(h)
```



Optimisation

- Calculates **optimal speeds** with wind:
  - Maximum range cruise (MRC)
  - Long range cruise (LRC)
  - Economic cruise speed (ECON)
  - Maximum endurance cruise (MEC)

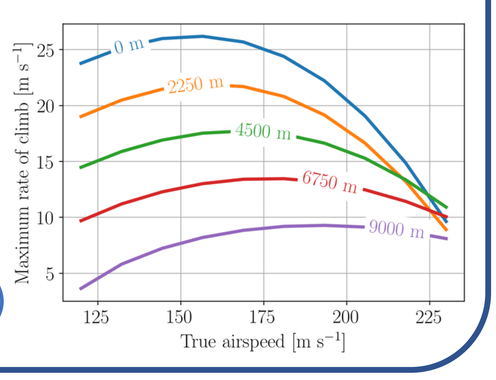
```
opt.MrcMach(aircraft,delta,theta,m,w)
>0.78
```



Visualisation

- Provides tools for visualisation of **aircraft trajectories**, **aircraft envelope** and **performance.**

```
visualisation.plot
(x=v, y=(Tmax-D)*v/(m*g), z=h)
```



Performance

- Automatically **parses** the BADA datasets.

```
aircraft = performance.bada3("XXX.OPF")
```

- Evaluates the **performance functions** of the BADA model at given flight conditions.

Numeric:
```
v = 447
T= 3.6
aircraft.ff(v=v, T=T)
> 3.39
```

Symbolic:
```
v = SX.sym('v')
T= SX.sym('T')
aircraft.ff(v=v, T=T)
> 0.94·(1 + v/1.0E+05)·T
```

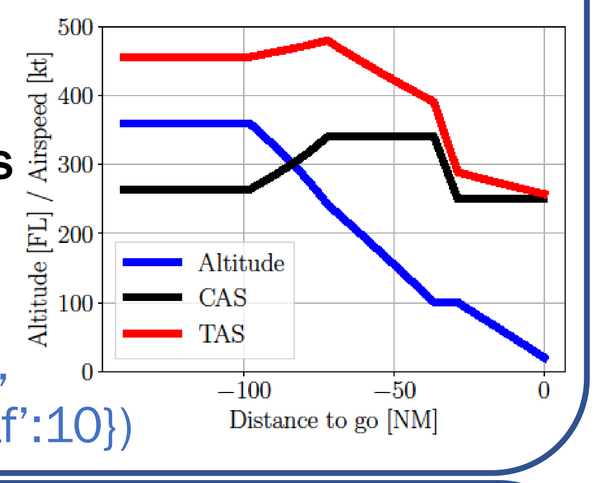
- Easy and fast implementation of the BADA APM and **derivatives** for **optimisation** and **sensitivity analysis.**

$$\mathrm{ECCF} = \frac{CI}{M\sqrt{\tau\gamma_a R}+w} + \frac{\delta\theta^{\frac{1}{2}} W_{mref} a_0 L_{HV}^{-1}}{M\sqrt{\tau\gamma_a R}+w} \sum_{i=0}^{4}\sum_{j=0}^{4} f_{5i+j+1} M^i \left(\frac{1}{2W_{mref}} p_0 k S M^2 \mathrm{K}\left(\left(d_1 + \frac{d_2}{(1-M^2)^{\frac{1}{2}}} + \frac{d_3}{(1-M^2)} + \frac{d_4}{(1-M^2)^{\frac{3}{2}}} + \frac{d_5}{(1-M^2)^2}\right) + \left(d_6 + \frac{d_7}{(1-M^2)^{\frac{3}{2}}} + \frac{d_8}{(1-M^2)^3} + \frac{d_9}{(1-M^2)^{\frac{9}{2}}} + \frac{d_{10}}{(1-M^2)^6}\right)\left(\frac{2mg}{\delta p_0 k S M^2}\right)^2 + \left(d_{11} + \frac{d_{12}}{(1-M^2)^7} + \frac{d_{13}}{(1-M^2)^{\frac{15}{2}}} + \frac{d_{14}}{(1-M^2)^8} + \frac{d_{15}}{(1-M^2)^{\frac{17}{2}}}\right)\left(\frac{2mg}{\delta p_0 k S M^2}\right)^6\right)\right)^j$$

Trajectory prediction

- Computes trajectories given the **initial conditions,** a sequence of **flight intents** and **weather conditions.**

```
xf = pred.predict(x0,intent,
                  mode, {'tf':10})
```



pyBADA has been used in...

- Real-time **optimal planning** and **guidance.**
- **Fuel estimation** from surveillance data.
- Generation of **realistic scenarios** for the assessment of wake-vortex hazards in en-route airspace.

SUPPORTED BY SESAR JOINT UNDERTAKING

Ramon Dalmau, Marc Melgosa & Xavier Prats – Techincal University of Catalonia
ramon.dalmau@upc.edu, marc.melgosa@upc.edu & xavier.prats@upc.edu