IAMOOC: exercises

Luc Jaulin

Exercise .0. *Hello intervals*

1) After having installed PyIbex and Vibes (see https://www.ensta-bretagne.fr/jaulin/pyibex_doc.pdf), write the PYTHON program below. You do not have to understand it. It is only to check that everything is properly installed.

```
from pyibex import *
from vibes import *
f = Function('x', 'y', 'x*cos(x-y)+y')
sep = SepFwdBwd(f, CmpOp.LEQ)
X0 = IntervalVector(2, [-10, 10] )
vibes.beginDrawing()
pySIVIA(X0, sep, 0.1)
vibes.saveImage('helloIntervals.jpg')
vibes.endDrawing()
```

Save the script into exo0.py.

2) Run the program in order to generate the image helloIntervals.jpg.

Exercise .1. Interval arithmetic

1) Recall that if $\diamond \in \{+, -, \cdot, /, \max, \min\}$, and if $[x]$ and $[y]$ are two intervals of $\mathbb{R}$ we have

$$[x] \diamond [y] \triangleq [\{x \diamond y \mid x \in [x], y \in [y]\}]$$

where $[\mathbb{A}]$ denotes the smallest interval which contains the set $\mathbb{A} \subset \mathbb{R}$. If $[x] = [-1, 3]$ and $[y] = [2, 5]$, compute $[x] \diamond [y]$ for $\diamond \in \{+, -, \cdot, /, \max, \min\}$.

2) Compute

$$\begin{array}{l} [-2,4] \cdot [1,3] \\ [-2,4] \sqcup [6,7] \\ \max([2,7],[1,9]) \\ \max(\emptyset,[1,2]) \\ [-1,3]/[0,\infty] \\ ([1,2] \cdot [-1,3]) + \max([1,3] \cap [6,7],[1,2]) \end{array}$$

where $\sqcup$ is the hull union, i.e., $[x] \sqcup [y] = [[x] \cup [y]]$.

3) If $f \in \{\text{sqr, sqrt, log, exp, } \dots\}$, is a function from $\mathbb{R}$ to $\mathbb{R}$, we define the interval extension as

$$f([x]) \triangleq [\{f(x) \mid x \in [x]\}].$$

Compute $\text{sqr}([-1, 3])$, $\text{sqrt}([-10, 4])$, $\log([-2, -1])$.

4) Compute

$$\begin{array}{l} ([1,2] + [-3,4]) \cdot [-1,5]. \\ \exp([1,2]/[0,\infty]). \end{array}$$

Exercise .2. Intervals with PYTHON

1) Under the PYTHON environment, implement the class `Interval` with basic elementary operations $+, -, \cdot, /$ and the functions `exp`, `log`, `sqr`, `min`, `max`. The resulting interval library will be called `myinterval.py`.

2) Check your library on the interval calculus of Exercise 1.

3) Consider the function

$$f(x) = x^2 + 2x - \exp x.$$

Using `myinterval.py`, compute a range for $f([x])$ where $[x] = [-2, 2]$.

Exercise .3. Minimization of a scalar valued function

1) Using VIBES, and using your own interval library `myinterval.py`, draw an interval enclosure for the function $f(x) = x^2 + 2x - \exp x$ where $x \in [-2, 2]$. For this purpose, you will have to build a PYTHON program, named `myminimizer.py`, which generates intervals enclosing $f([x](k))$ where $[x](k) = -2+\delta\cdot[k, k+1]$ for different values of δ. For instance, we may take $\delta \in \{0.5, 0.05, 0.005, 0.0005\}$.

2) From the previous question, give an interval containing the global minimum for f over $[-2, 2]$. Give also an interval containing the global minimizer.

3) Modify your program in order to compute automatically an interval containing the global minimum.

Exercise .4. Parameter estimation

Consider the bounded-error parameter estimation problem defined by

$$p_1 \cdot e^{p_2 \cdot t} \in [y]\,(t)$$

where $\mathbf{p} = (p_1, p_2)$ is the parameter vector, t is the time, $[y]\,(t)$ is the output interval that is returned by a sensor. Assume that for $t \in \mathbb{T} = \{0.2, 1, 2, 4\}$ we collected the following interval measurements:

$$\begin{array}{lll} [y]\,(0.2) & = & [1.5, 2] \\ [y]\,(1) & = & [0.7, 0.8] \\ [y]\,(2) & = & [0.1, 0.3] \\ [y]\,(4) & = & [-0.1, 0.03]\,. \end{array}$$

Define the set of all feasible parameter vectors as

$$\mathbb{P} = \left\{ \mathbf{p} \in [-3, 3]^2 \mid \forall t \in \mathbb{T},\, p_1 \cdot e^{p_2 \cdot t} \in [y]\,(t) \right\}.$$

where $[-3, 3]^2$ corresponds a prior box which is known to enclose $\mathbf{p}$.

1) Show that $\mathbb{P}$ corresponds to a set inversion problem.

2) Implement a library `mybox.py` with the class `Box` that is useful to the Sivia algorithm.

3) Using Python, Vibes and your own libraries `myinterval.py` and `mybox.py` compute and draw an inner and an outer approximation for $\mathbb{P}$. The corresponding Python will be named `myestimator.py`.

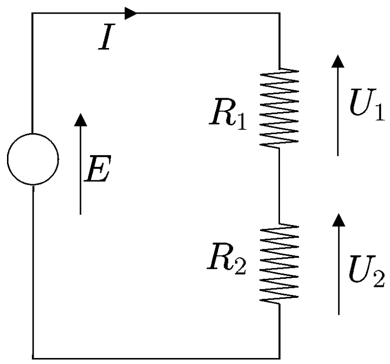
4) Modify your program to draw in the (y, t) space an enclosure of the set of all feasible outputs defined by

$$\mathbb{Y} = \left\{ (t, y) \mid \exists \mathbf{p} \in \mathbb{P},\, y = p_1 \cdot e^{p_2 \cdot t} \right\}.$$

5) Modify your program in order to characterize the set of all parameters $\mathbb{P}^{\{q\}}$ that are consistent with all data except q of them, where $q \in \{0, \ldots, 3\}$.

Exercise .5. Electric circuit

Consider the following electric circuit



We collected the following measurements:

$$\begin{aligned} E &\in [23V, 26V],\ I \in [4A, 8A],\ U_1 \in [10V, 11V], \\ U_2 &\in [14V, 17V],\ P \in [124W, 130W], \end{aligned}$$

where P is the power delivered by the battery.

1) Give the relations that link all these variables together.

2) With Python, program a contractor for the three constraints $z = x + y$, $z = x \cdot y$ and $y = x^2$.

3) Propose a contractor-based algorithm (stored into `mycircuit.py`) which computes two intervals $[R_1]$ and $[R_2]$ enclosing all feasible values for R_1 and R_2.

Exercise .6. Forward-Backward contractor

Given a point $\mathbf{c} = (c_x, c_y)$ which belongs to the box $[\mathbf{c}] = [1,3] \times [2,4] \subset \mathbb{R}^2$ and a real number $r \in [r] = [4,5]$. Consider the set

$$\mathbb{S} = \left\{(x,y) \in \mathbb{R}^2 \mid \exists \mathbf{c} \in [\mathbf{c}], \exists r \in [r],\ (x-c_x)^2 + (y-c_y)^2 = r^2\right\}.$$

1) With PYTHON, program a contractor $\mathcal{C}_{\mathbb{S}}$ for $\mathbb{S}$ using a forward-backward contraction algorithm.

2) Using a paver and the contractor $\mathcal{C}_{\mathbb{S}}$ inside a program that will be called `myring.py`, provide an outer approximation for $\mathbb{S}$.

Exercise .7. Separator

Given a set $\mathbb{S}$, a separator for $\mathbb{S}$ is a pair of two contractors: one for $\mathbb{S}$ and one for its complementary set $\overline{\mathbb{S}}$. More precisely, for a given box $[\mathbf{x}]$, a separator $\mathcal{S}$ for $\mathbb{S}$ will return two subboxes $[\mathbf{a}]$ and $[\mathbf{b}]$ of $[\mathbf{x}]$ such that $[\mathbf{a}] \cap \mathbb{S} = [\mathbf{x}] \cap \mathbb{S}$ and $[\mathbf{b}] \cap \overline{\mathbb{S}} = [\mathbf{x}] \cap \overline{\mathbb{S}}$. In this exercise, you have to return a PYTHON program named `myseparator.py`.

1) Using a the contractor developed in Exercise 6, program a separator for the set

$$\mathbb{S}_0 = \left\{ (x,y) \in \mathbb{R}^2 \mid \ (x - c_x)^2 + (y - c_y)^2 \in [r]^2 \right\}.$$

2) Consider the two rings defined by

$$\begin{array}{rcl} \mathbb{S}_1 & = & \left\{ (x,y) \in \mathbb{R}^2 \mid \ (x-1)^2 + (y-2)^2 \in [4,5]^2 \right\} \\ \mathbb{S}_2 & = & \left\{ (x,y) \in \mathbb{R}^2 \mid \ (x-2)^2 + (y-5)^2 \in [5,6]^2 \right\}. \end{array}$$

Compute a separator for $\mathbb{S}_1 \cap \mathbb{S}_2$ and draw an inner and an outer approximations for this intersection.

3) Compute a separator for $\mathbb{S}_1 \cup \mathbb{S}_2$ and draw an inner and an outer approximations for this union.

Exercise .8. Contractors and Separators with PyIbex

1) Using a the basic interval operation of PyIbex build a contractor for the set

$$\mathbb{S}_0 = \left\{(x,y) \in \mathbb{R}^2 \mid \exists\,(c_x,c_y) \in [1,3] \times [2,4]\,,\ (x-c_x)^2 + (y-c_y)^2 \in [2,4]^2\right\}.$$

Using a paver and Vibes, draw the corresponding paving. The program will be stored in `pyibexring.py`.

2) Repeat the previous question by using the statement `CtcFwdBwd` of PyIbex.

3) Consider the two rings defined by

$$\begin{array}{rcl}\mathbb{S}_1 & = & \left\{(x,y) \in \mathbb{R}^2 \mid \ (x-1)^2 + (y-2)^2 \in [4,5]^2\right\} \\ \mathbb{S}_2 & = & \left\{(x,y) \in \mathbb{R}^2 \mid \ (x-2)^2 + (y-5)^2 \in [5,6]^2\right\}.\end{array}$$

With PyIbex, compute a separator for $\mathbb{S}_1 \cap \mathbb{S}_2$ and draw an inner and an outer approximations for this intersection.

4) With PyIbex, compute a separator for $\mathbb{S}_1 \cup \mathbb{S}_2$ and draw the corresponding paving.

Exercise .9. Parameter estimation with PyIbex

This exercise a similar to Exercise 4, but we will use PyIbex for the resolution. Consider the bounded-error parameter estimation problem defined by

$$p_1 \cdot e^{p_2 \cdot t} \in [y](t)$$

where $\mathbf{p} = (p_1, p_2)$ is the parameter vector, t is the time, $[y](t)$ is the output interval that is returned by a sensor. Assume that for $t \in \mathbb{T} = \{t_1, t_2, t_3, t_4\} = \{0.2, 1, 2, 4\}$, we collected the following interval measurements:

$$\begin{array}{lll} [y](0.2) & = & [1.5, 2] \\ [y](1) & = & [0.7, 0.8] \\ [y](2) & = & [0.1, 0.3] \\ [y](4) & = & [-0.1, 0.03]\,. \end{array}$$

Define the set of all parameter vectors consistent with the ith data interval is defined by

$$\mathbb{P}_i = \left\{\mathbf{p} \in [-3, 3]^2 \ \mid p_1 \cdot e^{p_2 \cdot t_i} \in [y](t_i)\right\}$$

where $[-3, 3]^2$ corresponds a prior box which is known to enclose $\mathbf{p}$.

1) Using PyIbex and Vibes draw an inner and an outer approximations for each $\mathbb{P}_i$. The program will be called `pyibexestimator.py`.

2) For $q \in \{0, 1, 2\}$, compute the q-relaxed intersection

$$\mathbb{P}^{\{q\}} = \bigcap_{i}^{\{q\}} \mathbb{P}_i.$$

Discuss and compare with your results obtained for Exercise 4.

3) Assume that we may have outliers among the measurements. Provide a method able to identify which data intervals correspond to outliers.

Exercise .10. Localization of a robot using PyIbex

A robot has to localize inside an environment made of 4 landmarks $\mathbf{m}(i), i \in \{1, \dots, 4\}$. It is able to measure its distance $d(i)$ and the azimuth $\alpha(i)$ corresponding to each landmark with some bounded errors. The coordinates of landmarks and the interval measurements are given in the following table

i	$\mathbf{m}(i)$	$[d](i)$	$[\alpha](i)$
1	$(6,12)$	$[10,13]$	$[0.5,1]$
2	$(-2,-5)$	$[8,10]$	$\left[-3,-\frac{3}{2}\right]$
3	$(-3,10)$	$[5,7]$	$[1,2]$
4	$(3,4)$	$[6,8]$	$[2,3]$

The set of all positions $\mathbf{p} \in \mathbb{R}^2$ for the robot that are consistent with the ith measurements is

$$\mathbb{P}_i = \left\{(p_1, p_2) \in \mathbb{R}^2 \| \ \exists d \in [d](i), \exists \alpha \in [\alpha](i) \text{ s.t. } m_1(i) = p_1 + d\cos\alpha \text{ and } m_2(i) = p_2 + d\sin\alpha\right\}.$$

1) Using the optimal separator named SepPolarXY of PyIbex, draw a subpaving approximation for the sets $\mathbb{P}_i$.

2) Give the set of all $\mathbf{p}$ that are consistent with the largest number of data. The resulting program will be called `pyibexlocpie.py`.

3) Which data corresponds to an outlier.

Exercise .11. Simultaneous Localization and Mapping using PyIbex

Consider a robot at position (x, y) moving inside an unknown environment. We assume that its motion is described by the discrete time state equations:

$$\begin{cases} x(k+1) & = & x(k) + 10 \cdot \delta \cdot \cos\theta(k) \\ y(k+1) & = & y(k) + 10 \cdot \delta \cdot \sin\theta(k) \\ \theta(k+1) & = & \theta(k) + \delta \cdot (u(k) + n_u(k)) \end{cases}$$

where $k \in \{0, \ldots, 100\}$ is the discrete time, $\delta = 0.1$ corresponds to the sampling time, θ to the heading, u to the desired rotational speed and n_u to a noise. We assume that at time $k = 0$, we have $x(k) = y(k) = 0$ and $\theta(k) = 1$. The desired input $u(k)$ is chosen as

$$u(k) = 3 \cdot \sin^2(k\delta).$$

We assume that the heading is measured (using a compass for instance) with a small error:

$$\theta^m(k) = \theta(k) + n_\theta(k).$$

For all k we assume that $n_u(k)$ and $n_\theta(k)$ belong to $[-0.03, 0.03]$.

1) Simulate the system using uniform random noise and draw an interval tube enclosing the trajectory.

2) In the environment, we assume that we have 4 landmarks, the coordinate of which are given by the following table.

i	0	1	2	3
$\mathbf{m}(i)$	$(6, 12)$	$(-2, -5)$	$(-3, 20)$	$(3, 4)$

We do not know these coordinates. For each k the robot is able to measure the distance to one of these landmarks (taken randomly), with an accuracy of ± 0.03. By simulation, generate a set of intervals containing these all collected distances with their uncertainties.

3) Using a contractor-based method, improve the accuracy for the enclosure for the robot while simultaneously enclosing all landmarks by boxes. The resulting program will be stored in the file `pyibexSLAM.py`.