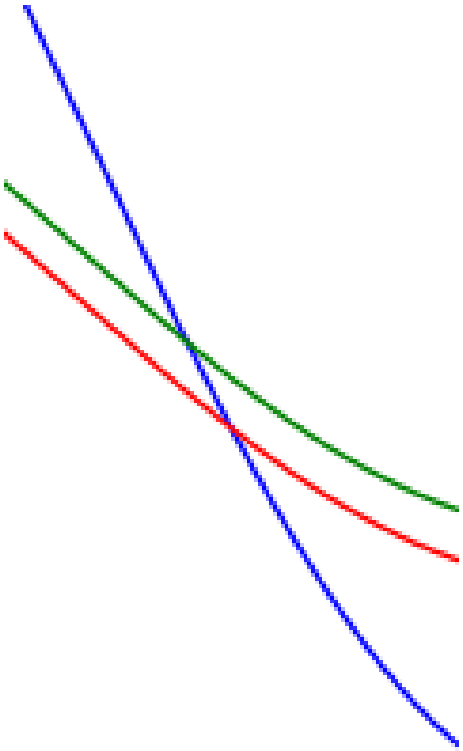


ISOPYBOX

A ready-to-use python stable isotope boxmodel

Jean-Alexis Hernandez, Théo Tacail

ISOPYBOX

A ready-to-use python stable isotope boxmodel

Jean-Alexis Hernandez, Théo Tacail

Introduction

Isopybox is a flexible stable isotope box model python code. The aim of this code is to provide a ready-to-use function allowing non coding persons to develop and test isotopic box models of their system of interest.

In short

Input

User fills an excel file[a] following a specific format with :

- Constants including : **reference material** isotopic true ratio, **run duration** in time units, number of wished **time steps** within this duration (...)
- List of **box names** together with **initial box sizes** (masses of X) and **initial isotopic compositions** (δ values)
- Matrix of **massic fluxes** of X between all boxes (mass unit per time unit)
- Matrix of **isotopic fractionation** coefficients between all boxes

Output

It allows the user to obtain **time evolution of isotope compositions and mass of element X** in each box **from initial state** to **final state** reached at the end of the specified run duration.

[a] provided template

This document aims at listing main pieces of information for you to be able to run the code on your computer.

NOTE

The isopybox function will soon be out of date.

The functionality of isopybox will soon be available as part of an integrated R package together with numerous other functions for dynamic stable isotope box modelling.

Contents

I Box modeling - Theory

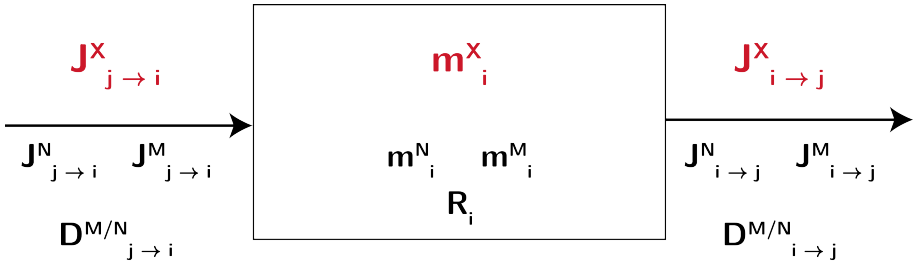
1 Principle of stable isotope box-model

We consider a system consisting of n boxes j, exchanging an element X that has[1] two isotopes *M* and *N*. Any box i is defined by :

m_i^X the mass of element X in box i - *we'll also call it box size afterwards*

m_i^M and m_i^N the masses of isotopes M and N in box i, respectively

$J_{i\mapsto j}^X$ is the massic flux of X from i to j (homogeneous to a mass per time unit)



We call R_i the M/N isotope ratio :

$$R_i = \left(\frac{n^M}{n^N}\right)_i = \left(\frac{m^M}{m^N}\right)_i \times \left(\frac{M^N}{M^M}\right) = \left(\frac{m^M}{m^N}\right)_i A$$

with:

- n_i^M and n_i^N the molar quantities of isotopes M and N in box i
- M^M and M^N the molar masses of isotopes M and N in box i
- $A = \left(\frac{M^N}{M^M}\right)$

Finally, we define $D_{i\mapsto j}^{M/N}$ the relative fractionation coefficient between M and N from i to j :

$$D_{i\mapsto j}^{M/N} = \frac{J_{i\mapsto j}^M / J_{i\mapsto j}^N}{m_i^M / m_i^N}$$

The principe of modeling the system consists now in predicting the evolution of **box sizes** during time and the evolution of the **isotope ratios**, knowing the fluxes, initial box sizes and fractionation coefficients.

2 Maths of stable isotope box modeling

a Mass conservation for X

Considering X has no radioactive nor radiogenic component, mass conservation for element X reads :

$$\frac{dm_i^X}{dt} = -\sum_{j\neq i} J_{i\mapsto j}^X + \sum_{j\neq i} J_{j\mapsto i}^X \tag{1}$$

This equation can be integrated considering constant fluxes as follows :

[1] at least

$$m_i^X = \left(-\sum_{j \neq i} J^X_{i \mapsto j} + \sum_{j \neq i} J^X_{j \mapsto i} \right) \times t + m^X_{i-initial} \tag{2}$$

b Differential equation fo R_i

Considering N and M are stable and non radiogenic, mass conservation reads :

$$\frac{dm_i^N}{dt} = -\sum_{j \neq i} J^N_{i \mapsto j} + \sum_{j \neq i} J^N_{j \mapsto i} = -\sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N} m_i^N + \sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N} m_j^N \tag{3}$$

Likewise,

$$\frac{dm_i^M}{dt} = -\sum_{j \neq i} \frac{J^M_{i \mapsto j}}{m_i^M} m_i^M + \sum_{j \neq i} \frac{J^M_{j \mapsto i}}{m_j^M} m_j^M \tag{4}$$

We can meanwhile derive the ratio R_i as :

$$\frac{dR_i}{dt} = \frac{d}{dt}\left(\frac{m^M}{m^N}\right)_i A = \frac{A}{{m_i^N}^2}\left(m_i^N \frac{dm_i^M}{dt} - m_i^M \frac{dm_i^N}{dt} \right) = \frac{A}{m_i^N}\left(\frac{dm_i^M}{dt} - \frac{R_i}{A}\frac{dm_i^N}{dt} \right)$$

We can insert *(3)* and *(4)* in previous relationship:

$$\frac{dR_i}{dt} = \frac{A}{m_i^N}\left(-\sum_{j \neq i} \frac{J^M_{i \mapsto j}}{m_i^M} m_i^M + \sum_{j \neq i} \frac{J^M_{j \mapsto i}}{m_j^M} m_j^M + \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N} m_i^N - \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N} m_j^N \right) \tag{5}$$

Modified as such :

$$\frac{dR_i}{dt} = A\left(-\sum_{j \neq i} \frac{J^M_{i \mapsto j}}{m_i^M}\frac{m_i^M}{m_i^N} + \sum_{j \neq i} \frac{J^M_{j \mapsto i}}{m_j^M}\frac{m_j^M}{m_j^N}\frac{m_j^N}{m_i^N} + \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N} - \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N}\frac{m_j^N}{m_i^N} \right)$$

$$\frac{dR_i}{dt} = A\left(-\sum_{j \neq i} \frac{J^M_{i \mapsto j}}{m_i^M}\frac{R_i}{A} + \sum_{j \neq i} \frac{J^M_{j \mapsto i}}{m_j^M}\frac{R_j}{A}\frac{m_j^N}{m_i^N} + \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N} - \frac{R_i}{A}\sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N}\frac{m_j^N}{m_i^N} \right)$$

$$\frac{dR_i}{dt} = R_i \sum_{j \neq i}\left(\frac{J^N_{i \mapsto j}}{m_i^N} - \frac{J^M_{i \mapsto j}}{m_i^M} \right) + \sum_{j \neq i} \frac{J^M_{j \mapsto i}}{m_j^M}\frac{m_j^N}{m_i^N} R_j - \sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N}\frac{m_j^N}{m_i^N} R_i$$

$$\frac{dR_i}{dt} = R_i \sum_{j \neq i}\left(\frac{J^N_{i \mapsto j}}{m_i^N} - \frac{J^M_{i \mapsto j}}{m_i^M} \right) + \sum_{j \neq i}\left(\frac{J^M_{j \mapsto i}}{m_j^M} R_j - \frac{J^N_{j \mapsto i}}{m_j^N} R_i \right)\frac{m_j^N}{m_i^N}$$

$$\frac{dR_i}{dt} = R_i \sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N}\left(1 - \frac{J^M_{i \mapsto j}/J^N_{i \mapsto j}}{m_i^M/m_i^N} \right) + \sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N}\left(\frac{J^M_{j \mapsto i}/J^N_{j \mapsto i}}{m_j^M/m_j^N} R_j - R_i \right)\frac{m_j^N}{m_i^N}$$

Resulting in :

$$\frac{dR_i}{dt} = R_i \sum_{j \neq i} \frac{J^N_{i \mapsto j}}{m_i^N}\left(1 - D^{M/N}_{i \mapsto j} \right) + \sum_{j \neq i} \frac{J^N_{j \mapsto i}}{m_j^N}\left(D^{M/N}_{j \mapsto i} R_j - R_i \right)\frac{m_j^N}{m_i^N} \tag{6}$$

In order to solve equation 6, we assumed that :

$$\frac{J^N_{i\mapsto j}}{m^N_i} \simeq \frac{J^X_{i\mapsto j}}{m^X_i} \quad ; \quad \frac{J^N_{j\mapsto i}}{m^N_j} \simeq \frac{J^X_{j\mapsto i}}{m^X_j} \quad ; \quad \frac{m^N_j}{m^N_i} \simeq \frac{m^X_j}{m^X_i}$$

We thus have approximated equation :

$$\frac{dR_i}{dt} = R_i \sum_{j\neq i} \frac{J^X_{i\mapsto j}}{m^X_i}\left(1 - D^{M/N}_{i\mapsto j}\right) + \sum_{j\neq i} \frac{J^X_{j\mapsto i}}{m^X_j}\left(D^{M/N}_{j\mapsto i} R_j - R_i\right)\frac{m^X_j}{m^X_i} \tag{7}$$

Equation 7 is the one used in the model for predicting evolution of isotope compositions in each box.

3 How does ISOPYBOX work ?

The ISOPYBOX programm :

- imports and reads the excel input file you filled
- calculates and displays the initial residence time for all boxes $\tau^X_{i_o} = \dfrac{m^X_{i_o}}{\sum_{j\neq i} J^X_{i\mapsto j}}$
- calculates the evolution of masses of X in each box at each time step
- integrates the derivative of $\dfrac{dR_i}{dt}$ at each time step using the **ode_int** function from the **scipy** package
- calculates $\delta^{M/N}X$ at each time step
- plots both *box sizes* and $\delta^{M/N}X$ as function of time

4 What ISOPYBOX can do

a Solve steady-state box model

When provided with balanced fluxes for all system boxes, provided that run duration is long enough, model can reach steady-state for isotopic compositions.

b Study response of system to discrete perturbation - relaxation

When intial state is known to be a steady state - from previous run for instance - it is possible to study response of system to a change of 1 parameter, such as a drastic change in one flux or isotope fractionation factor at a given time.

c Isotope ratio evolution with (some) unbalanced fluxes

When provided with (slightly) unbalanced fluxes for some boxes, model can predict evolution of isotope compositions for a given duration.

This implies that no box will be emptied. The user thus needs to pay attention at the residence times $\tau^X_{i-initial}$ which have to be higher in any box than total run duration. Otherwise unbalanced box model should be avoided.

II Run ISOPYBOX

1 Install python - Install spyder - Required Packages

The code should be able to run with both python 2.7 and python 3.5 (latest release).

The code is set to run with command lines only. Only a terminal should suffice to run it.

You can still use the spyder interface available within the anaconda distribution, although you don't need to see nor to modify the code. You can install python + anaconda (involving spyder) : https://www.continuum.io/downloads.

Two **python packages** will likely have to be installed on your computer :

- http://pandas.pydata.org/index.html.
- https://pypi.python.org/pypi/pydot
- If you want box sketches to be edited for each run, you will need

  http://www.graphviz.org/

  that might itself require https://www.macports.org/install.php for macs equal or later than El Capitan

2 Prepare input

a Design your model - draw a sketch

Before running any model, you need to properly define the boxes, fluxes, fractionation factors. Drawing a sketch helps.

Advice : ISOPYBOX does only enable modeling one close system. If you want to study an open system, you will have to design a model with a sub-system - consisting of the boxes representing your system of interest - **and** one or several infinite boxes.

For instance, for an open system such as an organism, **inputs** of the diet can be incoming from a extremely large reservoir compared to organism[2], which size and isotopic composition won't shift during the duration of the run.

On the other hand, for **outputs**, you can design an empty **waste** box that will be filled along the run, but that won't further interact with the sub-system, since it is a dead-end.

b Fill Excel input file

The design and setting of the model takes place in an excel file (**.xlsx or .xlsx**).

Its name has to be formated as follows : **Run_Name_INPUT.xlsx** - where you replace Run_Name with whatever you want but where **"_INPUT.xlsx" or "_INPUT.xls" remains.**

Following figures describe the way the excel _INPUT file should be filled.

[2] infinite

	CONSTS_ID	CONSTS
0	Element	Element symbol
1	Numerator	Mass number of numerator isotope
2	Denominator	Mass number of denominator isotope
3	Ratio_Standard	fill with reference material true isotope ratio (*e.g.* 56Fe/54Fe IRMM014)
4	time	wished duration of the run (time units - same as time units of fluxes)
5	n_steps	Wished number of calculation steps from 0 to time value

Figure 1: CONSTS sheet

BOXES_ID	SIZE_INIT	DELTA_INIT
box1	intial mass of element in each box (same mass unit as in fluxes)	intial $\delta^{M/N}X$ (‰) relative to reference material of each box
box2		
box3		
box4		
...	...	...

Figure 2: INITIAL sheet

	BOXES_ID	box1	box2	box3	box4	box5	...
0	box1	box1 --> box1	box1 --> box2	box1 --> box3	box1 --> box4	box1 --> box5	
1	box2	box2 --> box1	box2 --> box2	box2 --> box3	box2 --> box4	box2 --> box5	
2	box3	box3 --> box1	box3 --> box2	box3 --> box3	box3 --> box4	box3 --> box5	
3	box4	box4 --> box1	box4 --> box2	box4 --> box3	box4 --> box4	box4 --> box5	
4	box5	box5 --> box1	box5 --> box2	box5 --> box3	box5 --> box4	box5 --> box5	
...	...						

Fluxes in mass unit / time unit
(mass unit identical to sizes mass units
and
time unit identical to run time unit)

Figure 3: FLUXES sheet

	BOXES_ID	box1	box2	box3	box4	box5	...
0	box1	box1 --> box1	box1 --> box2	box1 --> box3	box1 --> box4	box1 --> box5	
1	box2	box2 --> box1	box2 --> box2	box2 --> box3	box2 --> box4	box2 --> box5	
2	box3	box3 --> box1	box3 --> box2	box3 --> box3	box3 --> box4	box3 --> box5	
3	box4	box4 --> box1	box4 --> box2	box4 --> box3	box4 --> box4	box4 --> box5	
4	box5	box5 --> box1	box5 --> box2	box5 --> box3	box5 --> box4	box5 --> box5	
...	...						

Fractionation coefficients

Figure 4: COEFFS sheet

Besides adding and naming boxes, you should not change sheet names, nor headers of columns in sheet CONSTS and INITIAL.

3 Run script

In order to run the script, you need to open the terminal of your computer and change directory to the folder containing ISOPYBOX.py. The command lines consists in designating the input file and the output folder as follows :

: $ python ISOPYBOX.py -i path to INPUT file -o path to INPUT **folder**

where text in blue refers to exact awaited command and text in red refers to commands depending on your simulation :

- the path to the excel **INPUT file**
- the path to the folder in which you want ISOPYBOX to save the **OUTPUT files**.

4 Read model output

Each run launched and performed implies creation of a **new OUTPUT folder** named after the date and time of the run.

This folder contains :

- a **Run_Name_OUTPUT.xlsx** file which is the Run_Name_INPUT.xlsx + a "**FINAL**" state sheet.
- a **Run_Name_Delta_t.pn**g figure of the evolution of deltas in all boxes with normal and logscale time
- a **Run_Name_Size_t.png** figure of the evolution of all box sizes with normal and logscaled time
- a **Run_Name_Delta_t.txt** text file of the evolution of deltas in all boxes with time - with tabulation separated columns
- a **Run_Name_Size_t.txt** text file of the evolution of all box sizes with time - with tabulation separated columns