Ordinary Differential Equations

as an alternative to agent-based modelling

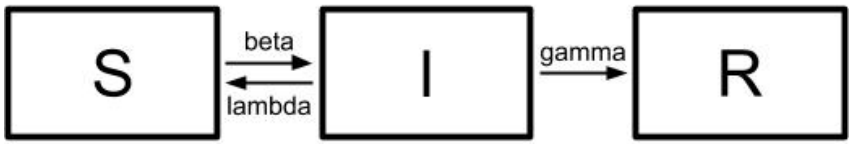
eX Modelo school

OpenMOLE

June 26, 2019

ODE systems

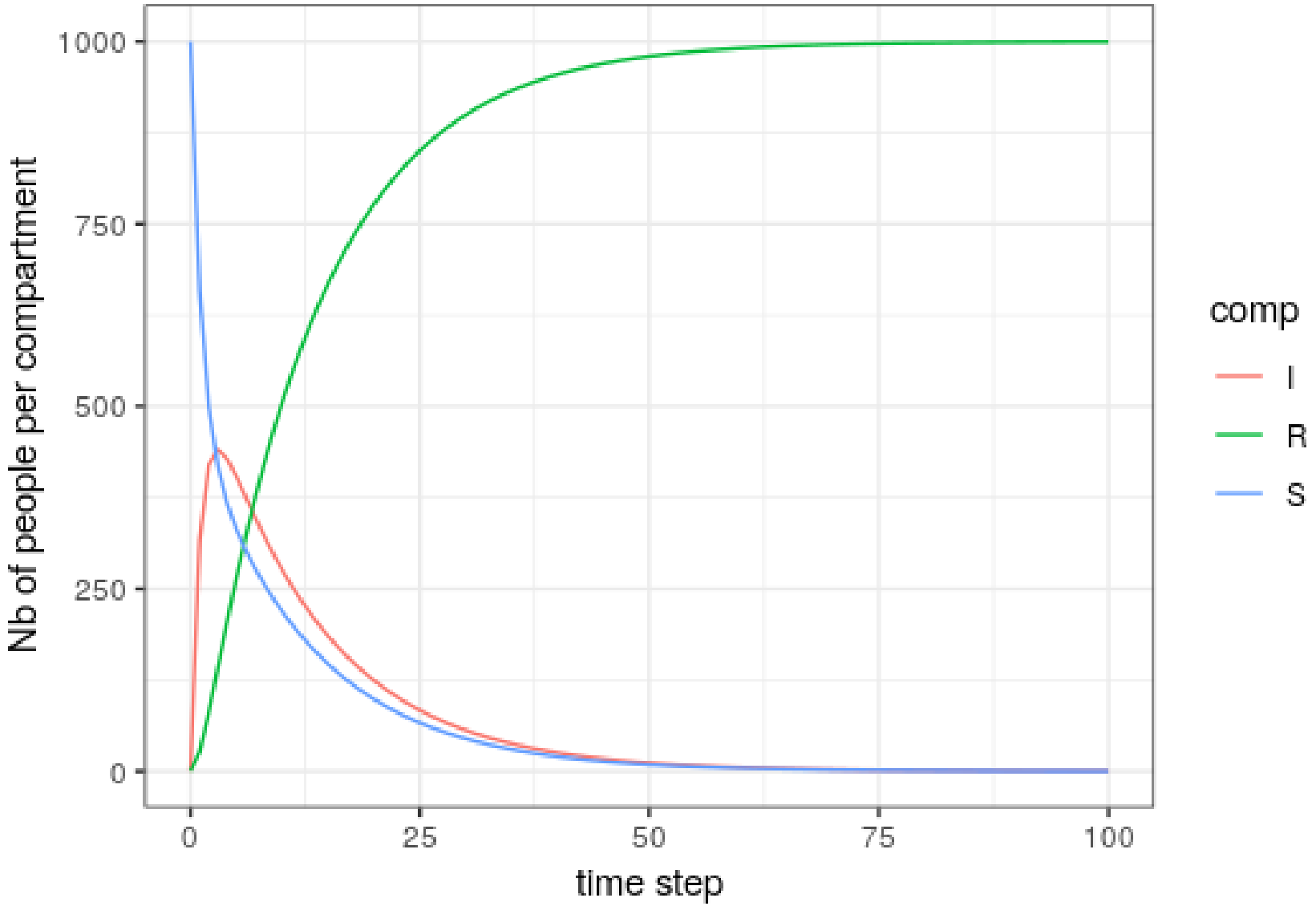
$\rightarrow$ widely used to model transmission phenomena



- population split into compartments
- system of ordinary differential equations

$$\begin{cases} \frac{\mathrm{d}S}{\mathrm{d}t} &= -\beta S + \lambda I \\ \frac{\mathrm{d}I}{\mathrm{d}t} &= \beta S - (\lambda + \gamma) I \\ \frac{\mathrm{d}R}{\mathrm{d}t} &= \gamma I \end{cases}$$

SIR model dynamics



ODE vs ABM

ODE	ABM
Equation-based	Individual-based
Generic mechanisms	Precise mechanisms
Population scale	Individual scale
Needs less resources	Computationally expensive

A Zombie situation

How could we model the Zombie invasion?

- Which mechanisms?
- Which parameters?

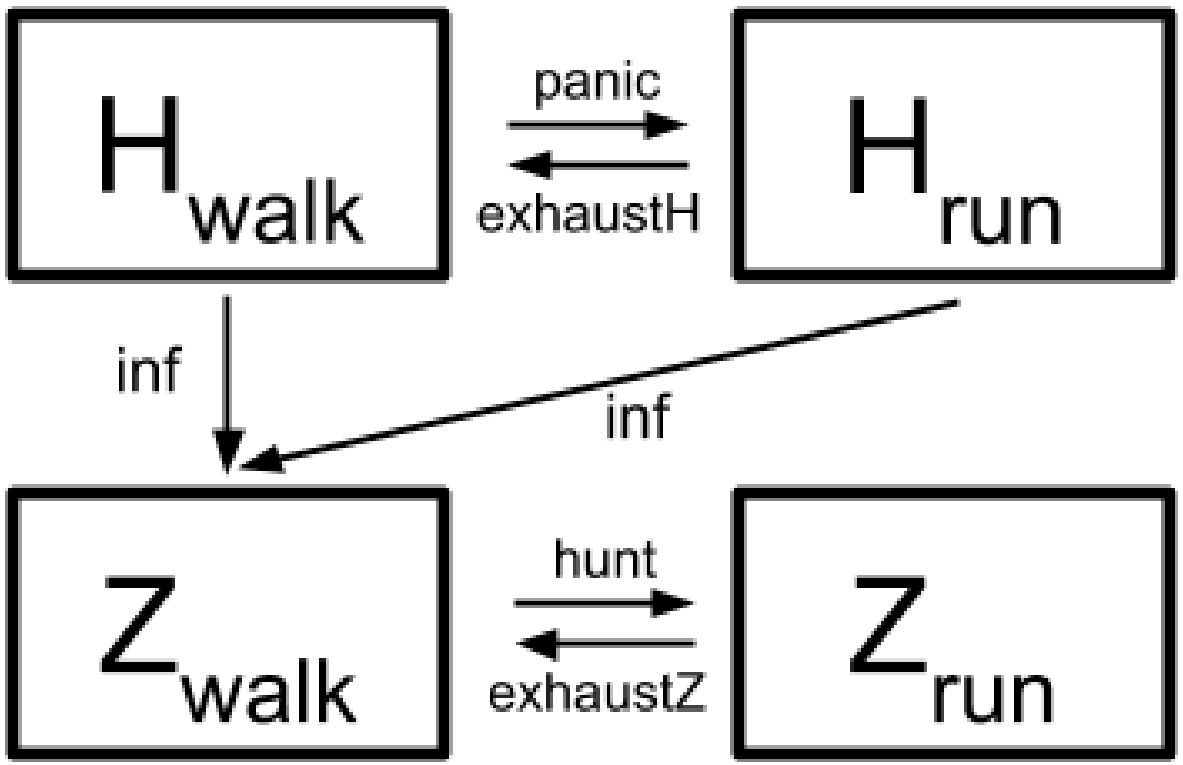
How could we model the Zombie invasion?

- Which mechanisms?
- Which parameters?

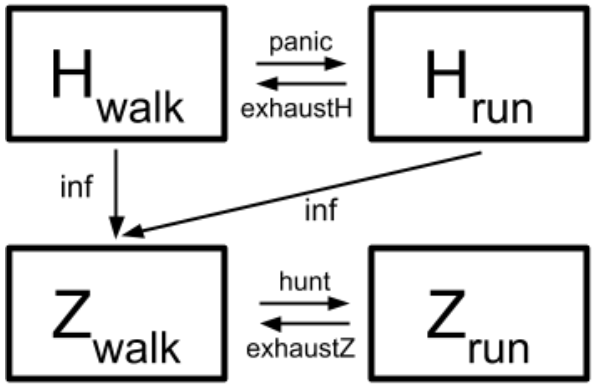
How can we assess our model's ability to reproduce the real data?

- Which metrics?
- Which fitness function?

A very simple ODE model



A very simple ODE model



$$
\begin{cases}
\dfrac{\mathrm{d}H_{walk}}{\mathrm{d}t} &= -(panic + inf) * H_{walk} + exhaustH * H_{run} \\
\dfrac{\mathrm{d}H_{run}}{\mathrm{d}t} &= panic * H_{walk} - (exhaustH + inf) * H_{run} \\
\dfrac{\mathrm{d}Z_{walk}}{\mathrm{d}t} &= inf * (H_{walk} + H_{run}) - hunt * Z_{walk} + exhaustZ * Z_{run} \\
\dfrac{\mathrm{d}Z_{run}}{\mathrm{d}t} &= hunt * Z_{walk} - exhaustZ * Z_{run}
\end{cases}
$$

Exploration

We have some real time series of zombie invasion
$\rightarrow$ find the parameter values to best fit them

We have some real time series of zombie invasion
$\rightarrow$ find the parameter values to best fit them

Process

First step: Calibrate

We have some real time series of zombie invasion
$\rightarrow$ find the parameter values to best fit them

Process

- Embed the model in OpenMOLE

First step: Calibrate

We have some real time series of zombie invasion
$\rightarrow$ find the parameter values to best fit them

Process

- Embed the model in OpenMOLE
- Define a fitness function

First step: Calibrate

We have some real time series of zombie invasion
$\rightarrow$ find the parameter values to best fit them

Process

- Embed the model in OpenMOLE
- Define a fitness function
- Write a calibration task

Parameter set

Second step: Profiles

Adding complexity

What mechanisms could we add to better represent the complexity of our Zombie situation?

What mechanisms could we add to better represent the complexity of our Zombie situation?

The parcimony issue

What mechanisms could we add to better represent the complexity of our Zombie situation?

The parcimony issue

- Do the new mechanisms really improve the fitness?

What mechanisms could we add to better represent the complexity of our Zombie situation?

The parcimony issue

- Do the new mechanisms really improve the fitness?
- Do we need them all?

What mechanisms could we add to better represent the complexity of our Zombie situation?

The parcimony issue

- Do the new mechanisms really improve the fitness?
- Do we need them all?
- What are the best combinations?

Study our model's parcimony

Process

- Embed the model in OpenMOLE **DONE**

Study our model's parcimony

Process

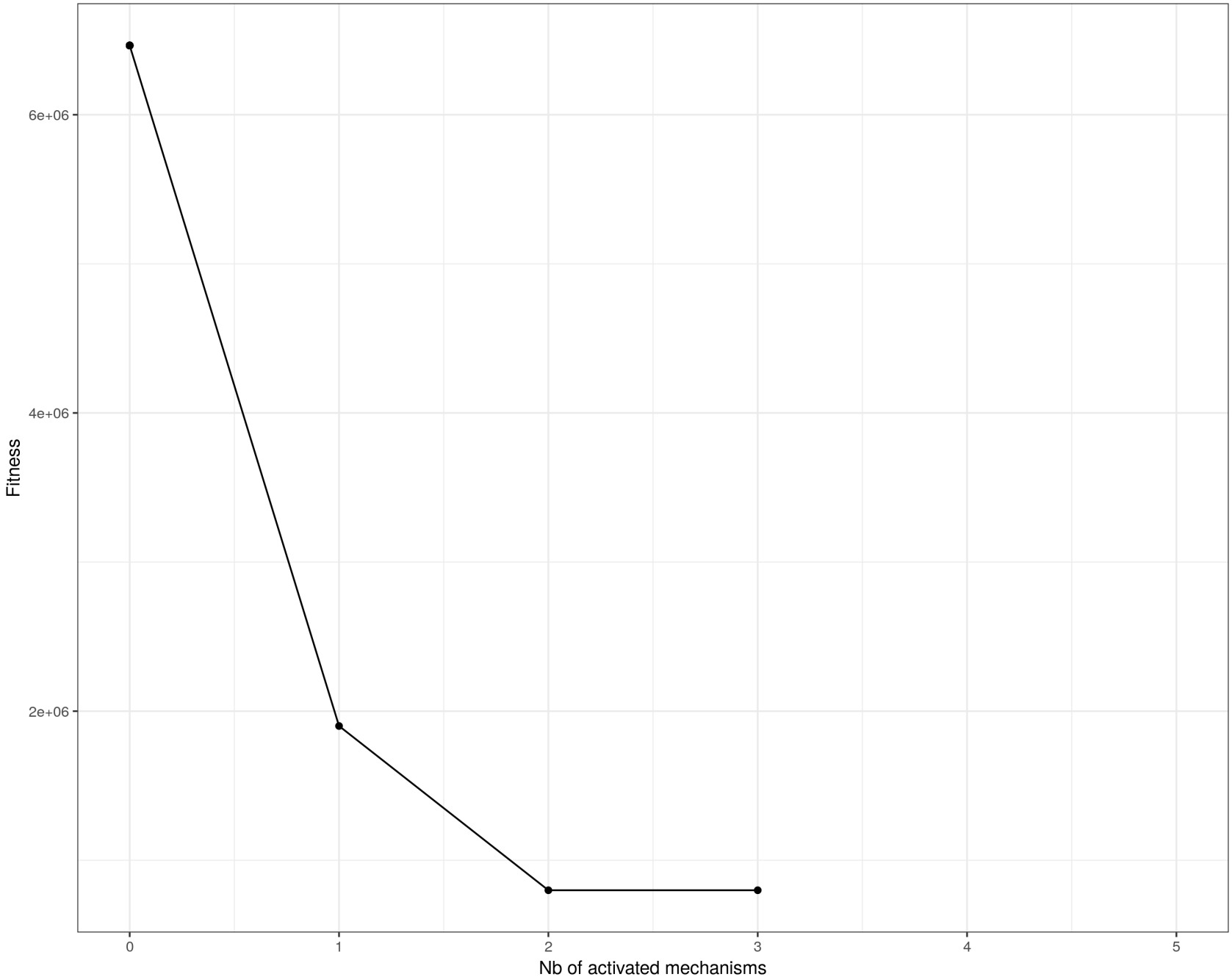
- Embed the model in OpenMOLE **DONE**
- Define a **second** fitness function

Study our model's parcimony

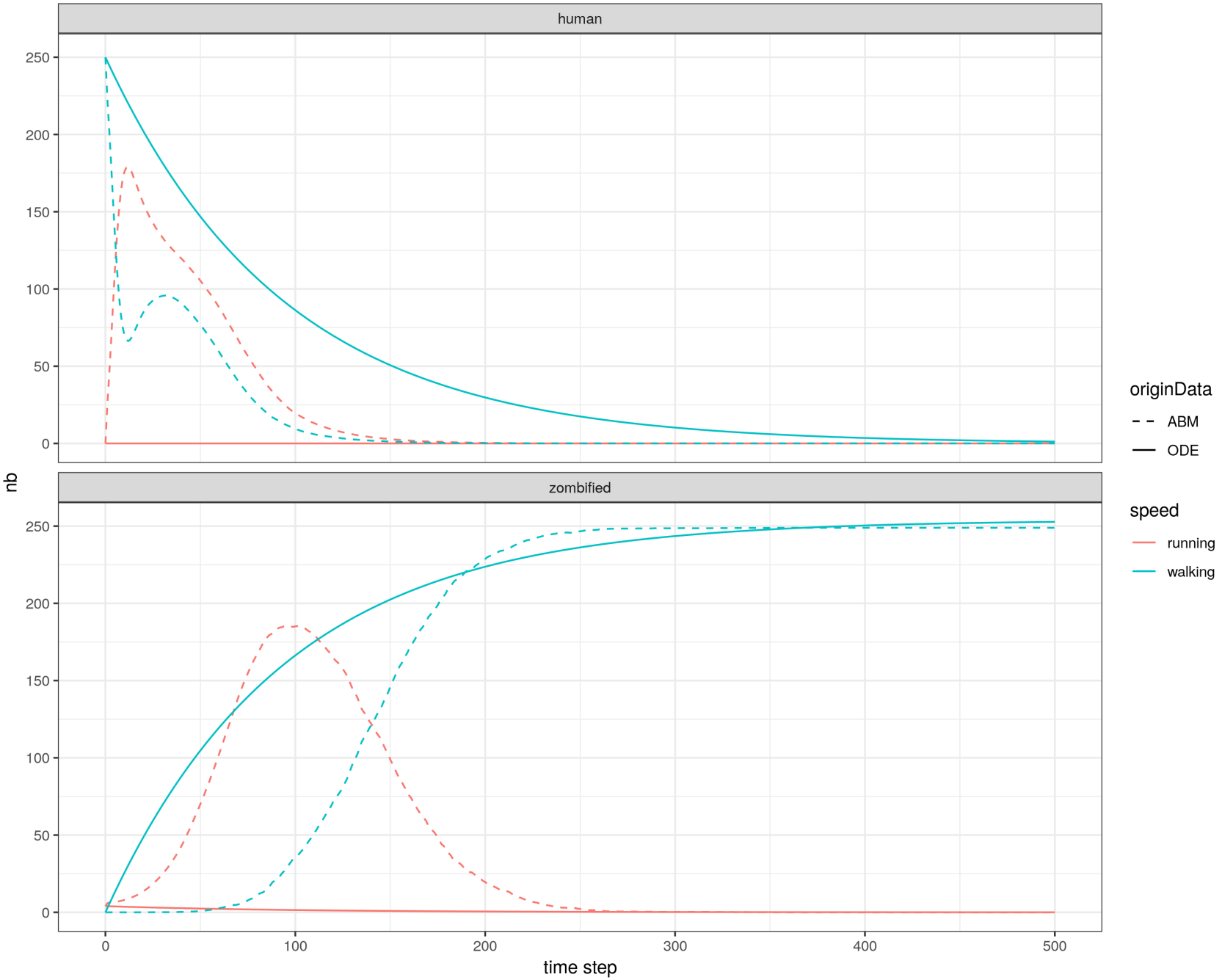
Process

- Embed the model in OpenMOLE **DONE**
- Define a **second** fitness function
- **Modify** the calibration task

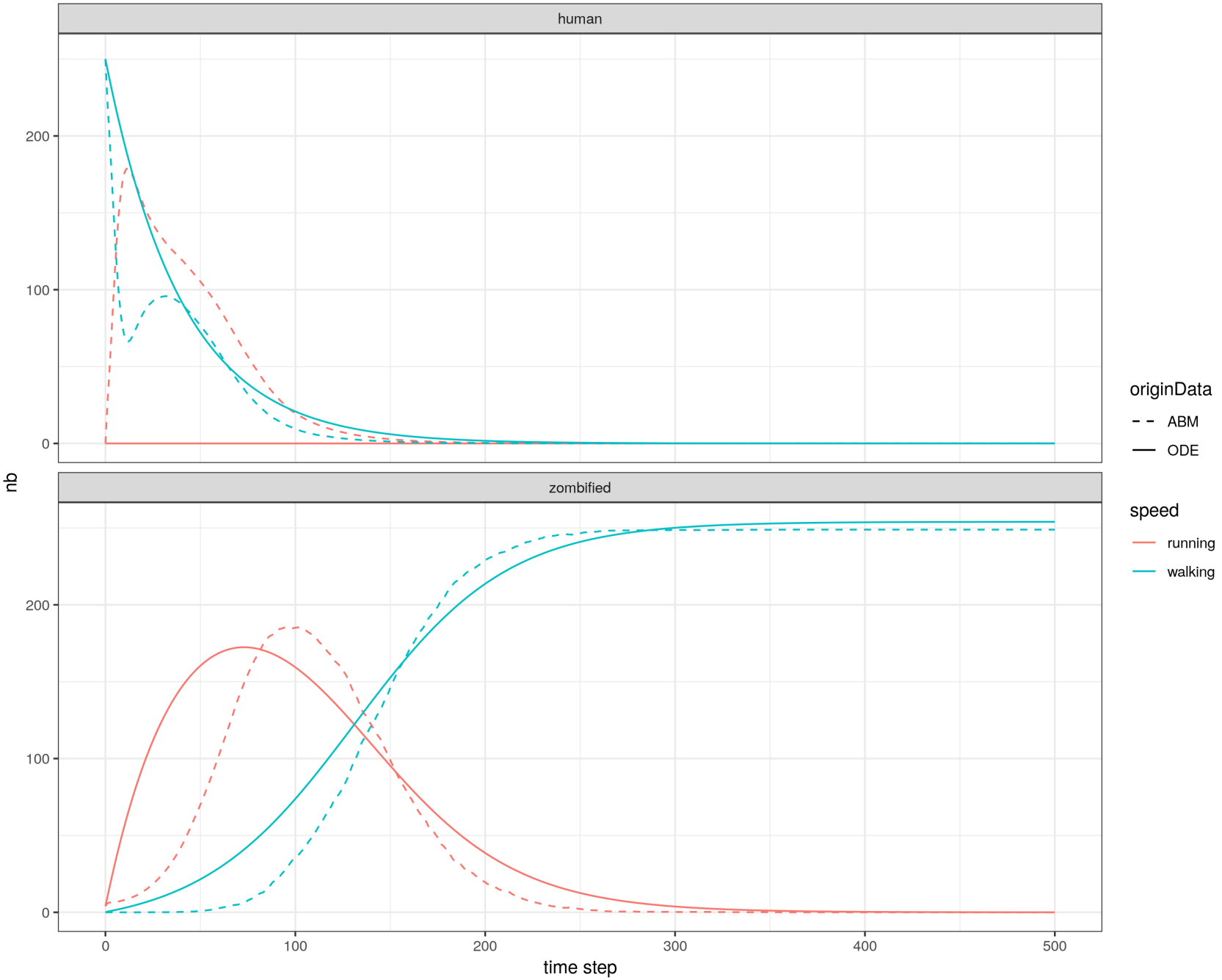
Pareto front



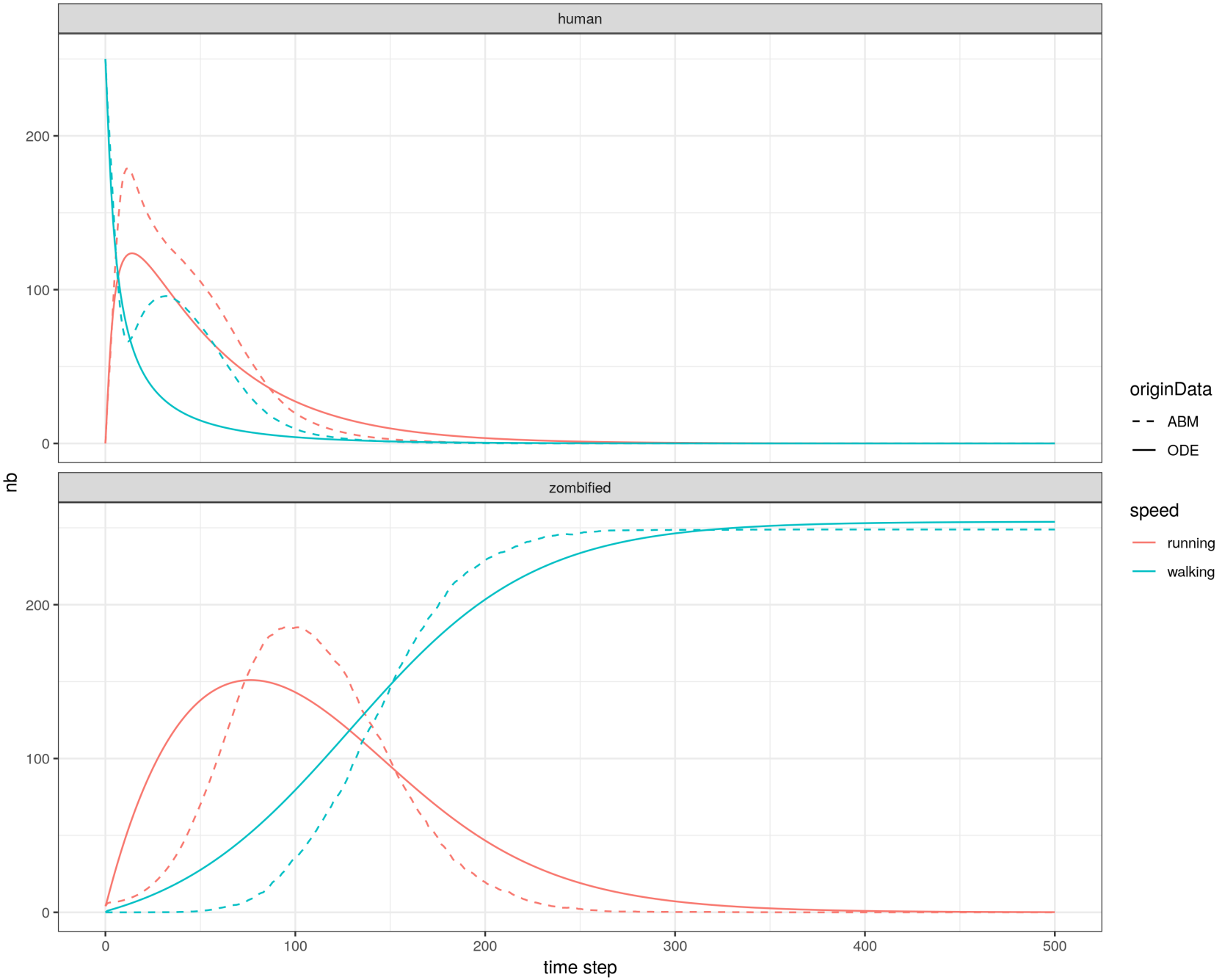
Dynamics for 0 mechanism activated



Dynamics for 1 mechanism activated



Dynamics for 2 mechanisms activated



Dynamics for 3 mechanisms activated