Did you say “effective size”?

On (some) methods that infer past Ne and how to interpret their outputs under structured models

Lounès Chikhi, Camille Steux

Centre de Recherche sur la Biodiversité et l’Environnement (CRBE) CNRS, Toulouse, France
Centre for ecology, evolution and environmental changes (ce3c) Univ. Lisboa, Portugal
International Research Laboratory (IRP) BEEG-B

Montpellier, DevOCGen, 4 Décembre 2025

Overview of the workshop

1. Introduction of the main concepts and inferential framework
   - 1.1. The PSMC
   - 1.2. The IICR (inverse instantaneous coalescence rate)
   - 1.3. SNIF (Structured Non-stationary Inference Framework)

2. SNIF in practice
   - 2.1. SNIF general workflow
   - 2.2. Getting started with SNIF
   - 2.3. Running SNIF on a PSMC file
   - 2.4. Refining the parameter space
   - 2.5. Validating the inferred scenarios
     - 2.5.1. About the validation step
     - 2.5.2. Extracting inferred scenarios as ms commands
     - 2.5.3. Running SNIF on a ms command
   - 2.6. Further validation of the inferred scenarios

3. General discussion on the concept of effective population size

practical parts

Introduction of the main concepts

SNIF in practice

SNIF
The piecewise stationary n-island model

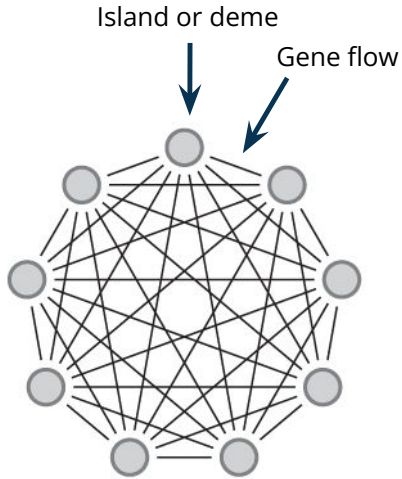
Time is divided into *c* time periods, called components, during which the parameters of the model are constant but can vary between components.

Parameters of a piecewise stationary n-island model:

- **n** - number of islands **constant**
- **N** - haploid deme size **constant**
- $\mathbf{M_i}$ - migration rate between t_i and t_{i+1} $= 2Nm_i$

  $\mathbf{t_i}$ - times at which M_i changes and with t_0 being the time of sampling (=0)

$i \in \{0, \ldots, c-1\}$



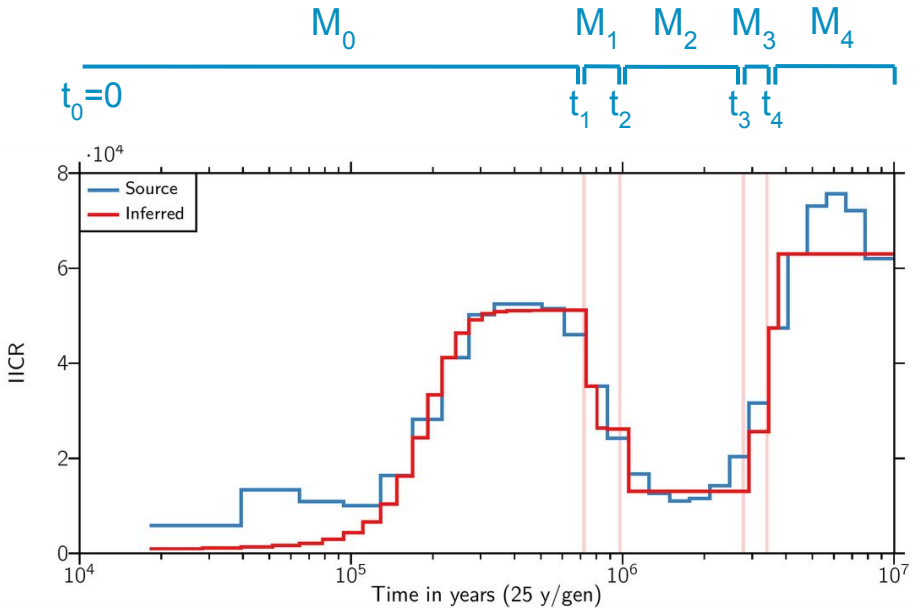
n-island model with n=9

The model is in units of haploids, but the parameters output by SNIF are in diploids

SNIF
Main principle

SNIF explores the parameter space of piecewise stationary n-island models and searches for the scenario that best fits the source PSMC or IICR.

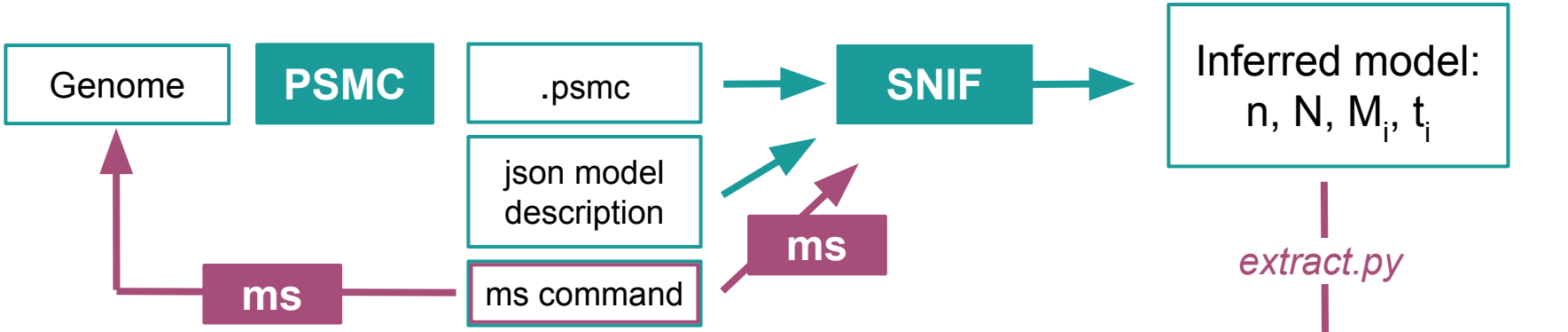
c is given by the user.



SNIF in practice

SNIF general workflow

- curves: csv file
- inferred parameters: csv file
- settings: json file



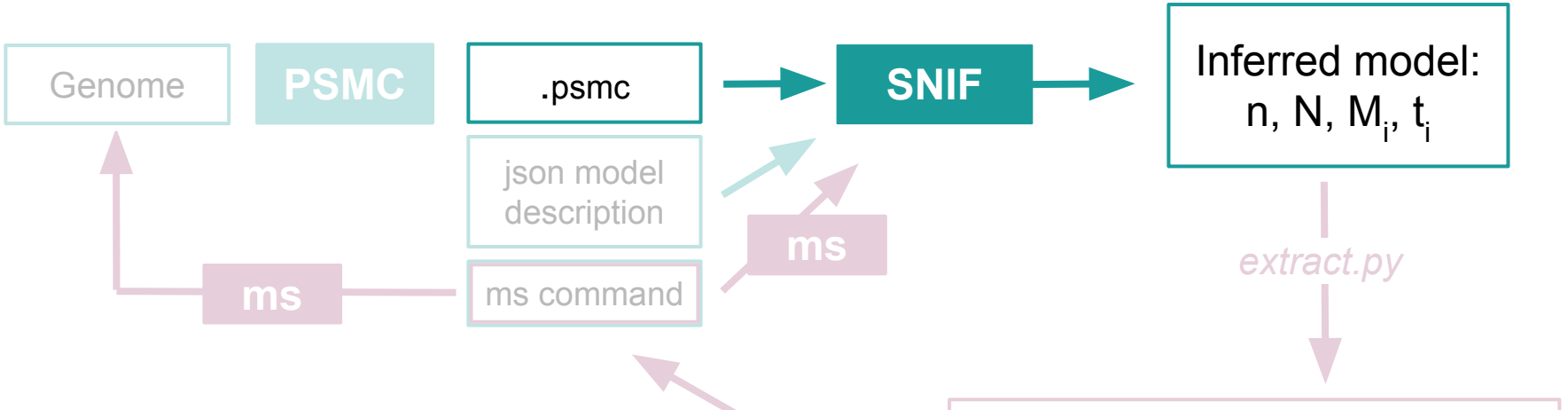
SNIF parameter files:

- c→nb of components
- w→distance weight
- n_{min}, n_{max} → range of nb of islands
- N_{min}, N_{max} → range of deme size
- …

SNIF in practice

SNIF general workflow

- curves: csv file
- inferred parameters: csv file
- settings: json file



SNIF parameter files:

- c→nb of components
- w→distance weight
- n_{min}, n_{max} → range of nb of islands
- N_{min}, N_{max} → range of deme size
- …

SNIF in practice
Getting started with SNIF

To run SNIF, you will need:

1. Some **input data** - examples can be found in ./workshop_files/input_data/. It can be:
    - a PSMC file: Pan_troglodytes_verus_Donald.psmc
    - a ms command that simulates:
        - T2: Pan_troglodytes_verus_Donald_inferred-model-004.ms
        - genomes: Pan_troglodytes_verus_Donald_inferred-model-004_10x100Mb.ms
    - a JSON file**:** Pan_troglodytes_verus_Donald_inferred-model-004.json

2. A **parameter file**, which is a python script - examples can be found in ./workshop_files/param_files/.

When both are ready:

- place the parameter file in the snif-main directory
- open a terminal in the snif-main directory
- launch SNIF using: `python3 param_file.py`

SNIF in practice
Getting started with SNIF

The parameter file contains:

1. the inference parameters:
   a. related to the input data
   b. related to the algorithm
   c. related to the parameter space
2. the settings parameters

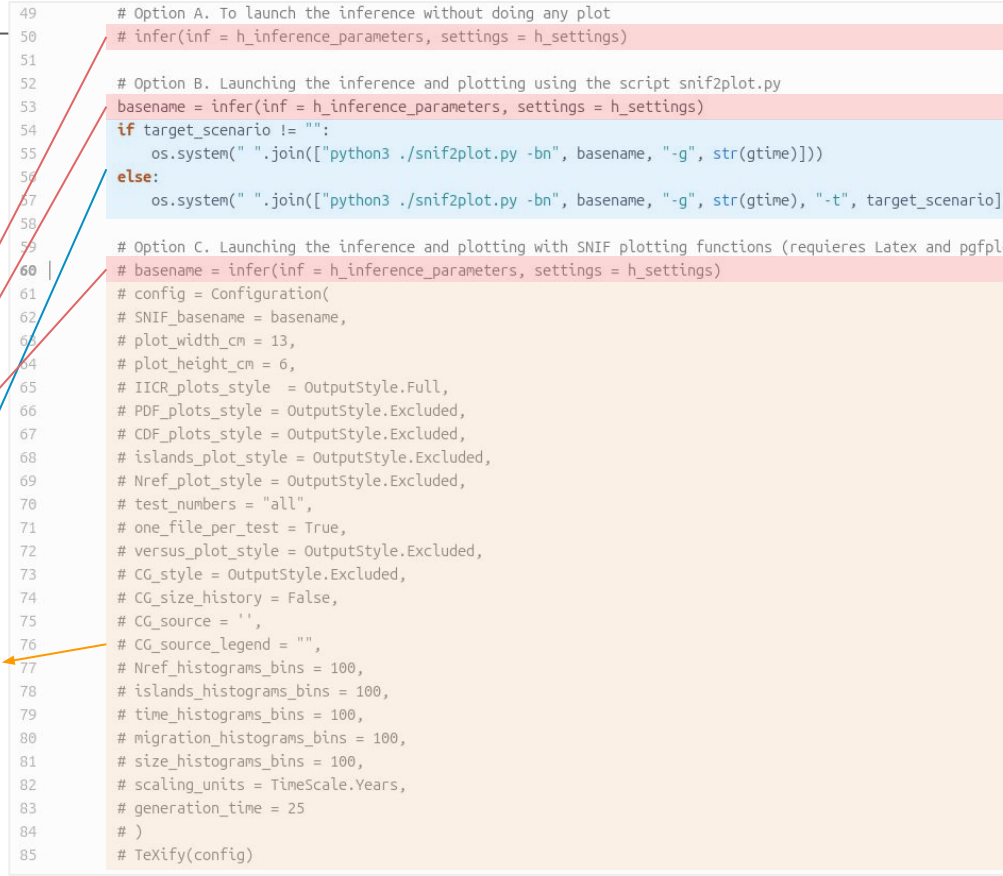
```
for c in [7]:
#for c in [5, 6, 7]:
    for w in [1]:
    #for w in [0.5, 1]:
        h_inference_parameters = InferenceParameters(
            data_source = './example_files/input_data/Pan_troglodytes_verus_Donald.psmc', #path to the input file
            source_type = SourceType.PSMC, #type of input (SourceType.PSMC, SourceType.MS or SourceType.JSON)
            IICR_type = IICRType.Exact, #type of IICR (IICRType.Exact if the input is a PSMC or JSON file, IICRType.T_sim i
genetic sequences)
            ms_reference_size = None, #ms diploid deme size
            ms_simulations = None, #number of coalescent times to simulate with ms
            psmc_mutation_rate = 1.5e-8, #mutation rate used to scale the input PSMC/IICR
            psmc_number_of_sequences = None, #number of ms simulated sequences
            psmc_length_of_sequences = None, #length of ms simulated sequences
            infer_scale = True, #whether to scale the results in generation time (True) or as T2 times (False)
            data_cutoff_bounds = (1e4/gtime, 2e7/gtime), #time period of the input data, in units of generations if infer_s
            data_time_intervals = 64, #time intervals of the psmc (parameter -p) / used to discretize the iicr
            distance_function = ErrorFunction.ApproximatePDF, #function used for the distance computation (ApproximatePDF f
            distance_parameter = w, #omega parameter to modulate the fitting (if omega<1 it gives less weight to the recent
            distance_max_allowed = 7e3, #threshold of the distance computed between the source and inferred IICR
            distance_computation_interval = (1e4/gtime, 2e7/gtime), #time period where the fitting will be computed, in uni
            rounds_per_test_bounds = (30, 30), #interval of the number of rounds (or iterations) of the algorithm
            repetitions_per_test = 1, #number of times to run SNIF (1 repetition corresponds to 1 inferred scenario)
            number_of_components = c, #number of components
            bounds_islands = (2, 100), #range values for the number of demes
            bounds_migrations_rates = (0.05, 50), #range values for the migration rates
#           bounds_migrations_rates = [(0.05, 50), (0.05, 50), (0.05, 50), (0.05, 50), (0.05, 50), (0.05, 50), (0.05, 50)],
            bounds_deme_sizes = (1, 1), #range values for the size ratio (keep (1,1) for constant deme size)
#           bounds_deme_sizes = [(1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1), (1, 1)],
            bounds_event_times = (1e4/gtime, 2e7/gtime), #time window(s) during which demographic events can occur, in unit
#           bounds_event_times = [(1.7e4/gtime, 3e6/gtime), (1.7e4/gtime, 3e6/gtime), (1.7e4/gtime, 3e6/gtime), (1.7e4/gti
            bounds_effective_size = (10, 10000) #range values for the deme size, in numbers of diploids
            )

        h_settings = Settings(
            static_library_location = './libs/libsnif.so', #path to the library files, not to change
            custom_filename_tag = 'Ptverus_Donald_frompsmc', #custom tag added to the output name
            output_directory = './example_files/SNIF_results', #path to the output directory
            default_output_dirname = './SNIF_results_default' #default output directory, created if output_directory is not
            )
```

SNIF in practice
Getting started with SNIF

The parameter file contains:

1. the inference parameters:
   a. related to the input data
   b. related to the algorithm
   c. related to the parameter space
2. the settings parameters
3. the command that launches the inference
4. the plotting related parameters:
   a. to plot with the python script ./snif2plot.py
   b. to plot with the dedicated function in SNIF



```
# Option A. To launch the inference without doing any plot
# infer(inf = h_inference_parameters, settings = h_settings)

# Option B. Launching the inference and plotting using the script snif2plot.py
basename = infer(inf = h_inference_parameters, settings = h_settings)
if target_scenario != "":
    os.system(" ".join(["python3 ./snif2plot.py -bn", basename, "-g", str(gtime)]))
else:
    os.system(" ".join(["python3 ./snif2plot.py -bn", basename, "-g", str(gtime), "-t", target_scenario]

# Option C. Launching the inference and plotting with SNIF plotting functions (requieres Latex and pgfpl
# basename = infer(inf = h_inference_parameters, settings = h_settings)
# config = Configuration(
# SNIF_basename = basename,
# plot_width_cm = 13,
# plot_height_cm = 6,
# IICR_plots_style  = OutputStyle.Full,
# PDF_plots_style = OutputStyle.Excluded,
# CDF_plots_style = OutputStyle.Excluded,
# islands_plot_style = OutputStyle.Excluded,
# Nref_plot_style = OutputStyle.Excluded,
# test_numbers = "all",
# one_file_per_test = True,
# versus_plot_style = OutputStyle.Excluded,
# CG_style = OutputStyle.Excluded,
# CG_size_history = False,
# CG_source = '',
# CG_source_legend = "",
# Nref_histograms_bins = 100,
# islands_histograms_bins = 100,
# time_histograms_bins = 100,
# migration_histograms_bins = 100,
# size_histograms_bins = 100,
# scaling_units = TimeScale.Years,
# generation_time = 25
# )
# TeXify(config)
```

SNIF in practice

Running SNIF on a PSMC file

Let us run SNIF on a PSMC file. The input file can found at:
./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc.
The corresponding parameter file can be found at:
./workshop_files/param_files/param_Ptroglodytes_verus_Donald_psmc.py

SNIF in practice
Running SNIF on a PSMC file

Let us run SNIF on a PSMC file. The input file can found at:
./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc.
The corresponding parameter file can be found at:
./workshop_files/param_files/param_Ptroglodytes_verus_Donald_psmc.py

1. Copy and paste the parameter file in snif-main
2. Open a terminal in snif-main
3. Run SNIF with: `python3 param_Ptroglodytes_verus_Donald_psmc.py`

```
(spyder-env) csteux@PP04018:~/Documents/Doctorat/SNIF/workshop_devocgen/snif-main$ python3 param_Ptroglodytes_verus_Donald_psmc.py

Welcome to SNIF -- an Inferential Framework for Demographic Structure
found 1 source file of type PSMC
loading static library..
initializing static library..
D Runtime initialized succesfully

loading file 1 of 1: ./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc
repetition 1 of 2
  round 1 of 30
  round 2 of 30
```

SNIF in practice
Running SNIF on a PSMC file

You should have **5 files** created in ./workshop_files/SNIF_results/:

- **basename_frompsmc.csv** = inferred parameters and other inference statistics
- **basename_frompsmc_curves.csv** = the source PSMC and inferred IICR
- **basename_frompsmc_settings.json** = summary of the parameters used for the inference

The plots were created using the script snif2plot.py, and therefore you should also have:

- **basename_frompsmc_IICR.pdf** ➜ shows the fit of the inferred IICR to the input PSMC curve
- **basename_frompsmc_CG.pdf** ➜ shows the inferred demographic parameters

Note that if several repetitions were done with SNIF, all the inferred scenarios are in the same output files.

SNIF in practice

Running SNIF on a PSMC file

- **basename_frompsmc.csv** = inferred parameters and other inference statistics

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	AA	AB
1	SEP=																											
2	id	d_omega	d_visual	rounds	evaluations	failed evals	duration	inf. n	inf. t0	inf. t1	inf. t2	inf. t3	inf. t4	inf. t5	inf. M0	inf. M1	inf. M2	inf. M3	inf. M4	inf. M5	inf. s0	inf. s1	inf. s2	inf. s3	inf. s4	inf. s5	inf. N_ref	source filename
3	1	946993	362777	30	5953640	0	153.694	46	0	7190	9160	16760	36406	1E+05	0.266849	5.4365	49.5522	0.532951	9.45385	0.231776	1	1	1	1	1	1	253.482	./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc
4	2	818628	317916	30	5939640	0	310.289	53	0	6581	8626.5	18573	36412	1E+05	0.226261	1.06633	47.6319	0.504429	10.3248	0.218987	1	1	1	1	1	1	211.821	./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc
5																												
6																												
7																												

Repetition ID

Distance using the weighted approach

# of rounds

Distance using the visual approach

Inferred n

Inferred t_i

Inferred M_i

Inferred size change

Inferred N

path to the source data

SNIF in practice

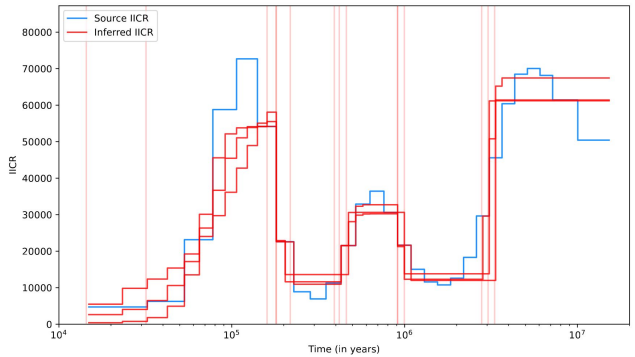
Running SNIF on a PSMC file

- **basename_frompsmc_curves.csv** = the source PSMC and inferred IICR

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	time-1	400	595.633	932.579	1298.68	1696.42	2128.53	2598.03	3108.1	3662.26	4264.33	4918.44	5629.12	6401.25	7240.12	8151.5	9141.67	10217.4	11386.2	12656	14035.6	15534.5	17162.9
2	source_iicr-1	4721.69	4721.69	4721.69	6249.57	6249.57	23149.1	23149.1	58798.3	58798.3	72690.9	72690.9	54162.3	54162.3	22602	22602	8884.02	8884.02	6926.73	6926.73	11267.6	11267.6	21532.3
3	source_cdf-1	0.000211789	0.0405862	0.106665	0.165442	0.2169	0.250538	0.265585	0.276776	0.28356	0.290164	0.296523	0.304531	0.314375	0.332173	0.358566	0.406478	0.474166	0.547477	0.623275	0.679234	0.719188	0.748458
4	source_pdf-1	0.000211744	0.000203193	0.000189198	0.000133538	0.000125305	0.0000323755	0.0000317255	0.0000123001	0.0000121847	0.0000976513	0.0000967765	0.0000128405	0.0000126587	0.0000295473	0.0000283795	0.0000668078	0.0000591887	0.0000653299	0.0000543871	0.000028468	0.0000249221	0.0000116821
5	inferred_iicr-1	344.864	435.555	742.506	1531.2	3643.29	9197.67	21358.5	37822.8	48878.8	52919.4	53949.2	54162.3	54199.8	22602	13719.6	13719.6	11885.5	11885.5	11885.5	11885.5	11885.5	21532.3
6	inferred_cdf-1	0.498922	0.611393	0.713173	0.759731	0.779779	0.788084	0.791636	0.793463	0.794763	0.795963	0.797205	0.798533	0.799964	0.801576	0.807845	0.814655	0.822828	0.831328	0.840102	0.849118	0.858338	0.867116
7	inferred_pdf-1	0.00145297	0.000892211	0.000386296	0.000156915	0.0000604455	0.0000230401	0.00000975556	0.00000546065	0.0000041989	0.00000385561	0.00000375899	0.00000371968	0.00000369072	0.00000877905	0.0000140058	0.0000135095	0.0000149066	0.0000141914	0.0000134532	0.0000126946	0.0000119189	0.00000617137
8	model_iicr-1																						
9	model_cdf-1																						
10	model_pdf-1																						
11	time-2	400	595.633	932.579	1298.68	1696.42	2128.53	2598.03	3108.1	3662.26	4264.33	4918.44	5629.12	6401.25	7240.12	8151.5	9141.67	10217.4	11386.2	12656	14035.6	15534.5	17162.9
12	source_iicr-2	4721.69	4721.69	4721.69	6249.57	6249.57	23149.1	23149.1	58798.3	58798.3	72690.9	72690.9	54162.3	54162.3	22602	22602	8884.02	8884.02	6926.73	6926.73	11267.6	11267.6	21532.3
13	source_cdf-2	0.000211789	0.0405862	0.106665	0.165442	0.2169	0.250538	0.265585	0.276776	0.28356	0.290164	0.296523	0.304531	0.314375	0.332173	0.358566	0.406478	0.474166	0.547477	0.623275	0.679234	0.719188	0.748458
14	source_pdf-2	0.000211744	0.000203193	0.000189198	0.000133538	0.000125305	0.0000323755	0.0000317255	0.0000123001	0.0000121847	0.0000976513	0.0000967765	0.0000128405	0.0000126587	0.0000295473	0.0000283795	0.0000668078	0.0000591887	0.0000653299	0.0000543871	0.000028468	0.0000249221	0.0000116821
15	inferred_iicr-2	296.888	390.916	746.812	1796.15	5009.42	14193.3	32595.2	50147.1	57515	59332.6	59673.9	59727.2	59734.3	22019.1	21460.5	11453.5	11453.5	11453.5	11453.5	11453.5	11453.5	11453.5
16	inferred_cdf-2	0.559368	0.670266	0.761191	0.79749	0.81105	0.815982	0.817969	0.819081	0.819998	0.820919	0.8219	0.822957	0.824098	0.826531	0.830156	0.835732	0.843268	0.851065	0.859096	0.867332	0.875735	0.884262
17	inferred_pdf-2	0.00148417	0.00084349	0.000319772	0.000112747	0.000037719	0.0000129651	0.00000558461	0.00000360776	0.00000312965	0.00000301825	0.00000298455	0.00000296419	0.00000294475	0.00000787813	0.00000791428	0.0000143422	0.0000136842	0.0000130035	0.0000123023	0.0000115833	0.0000108496	0.0000101051
18	model_iicr-2																						
19	model_cdf-2																						

To plot the source and inferred (best) IICR:

- time-i (x)
- source-iicr-i (y)
- inferred-iicr-i (y)



SNIF in practice
Running SNIF on a PSMC file

- **basename_frompsmc_settings.json** = summary of the parameters used for the inference


```
{
    "inference_parameters": {
        "data_source": "./workshop_files/input_data/Pan_troglodytes_verus_Donald.psmc",
        "source_type": "PSMC",
        "IICR_type": "Exact",
        "infer_scale": true,
        "data_cutoff_bounds": [
            400.0,
            800000.0
        ],
        "data_time_intervals": 64,
        "distance_function": "ApproximatePDF",
        "distance_parameter": 0.5,
        "distance_max_allowed": 7000.0,
        "distance_computation_interval": [
            400.0,
            800000.0
        ],
        "rounds_per_test_bounds": [
            30,
            30
        ],
        "repetitions_per_test": 2,
        "number_of_components": 6,
        "bounds_islands": [
            2,
            100
        ],
        "bounds_migrations_rates": [
            0.05,
            50
        ],
        "bounds_deme_sizes": [
            1,
            1
        ],
        "bounds_event_times": [
            400.0,
            800000.0
        ],
        "bounds_effective_size": [
            10,
            10000
        ],
        "psmc_mutation_rate": 1.5e-08,
        "psmc_number_of_sequences": null,
```

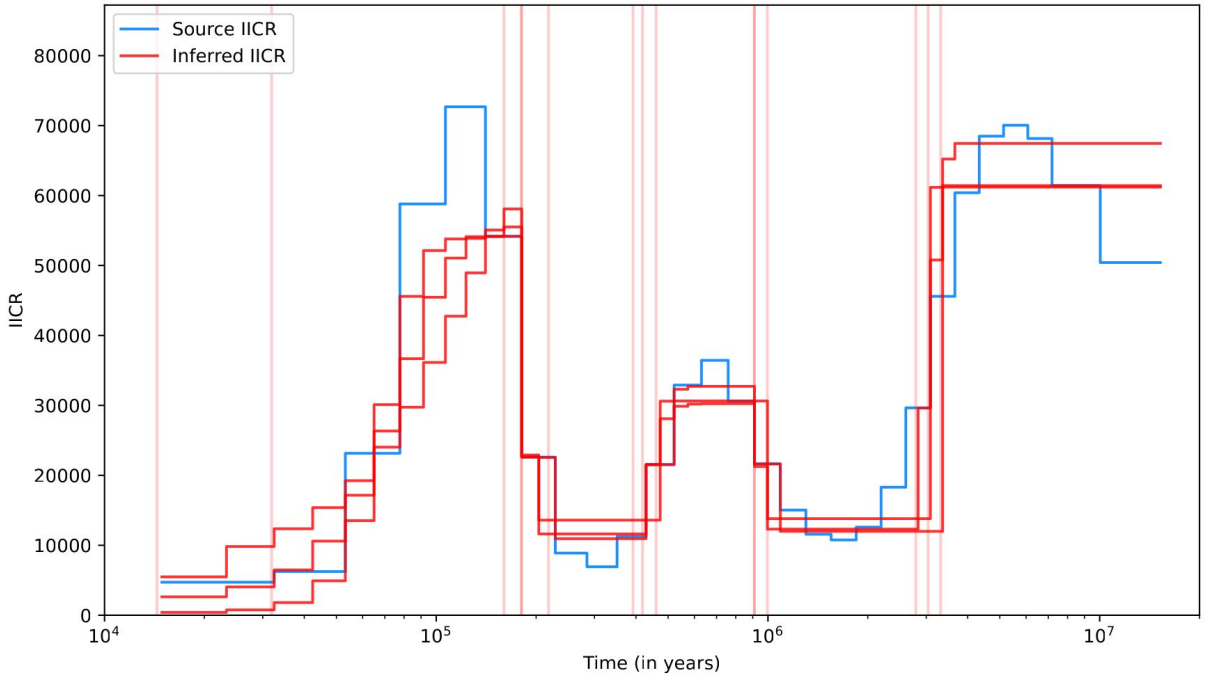

Note that this file is created at the end of the inference, and is created only if the inference went through.

SNIF in practice
Running SNIF on a PSMC file

- **basename_frompsmc_IICR.pdf** ➜ shows the fit of the inferred IICR to the input PSMC curve

c = 6
w = 0.5
3 repetitions
30 rounds

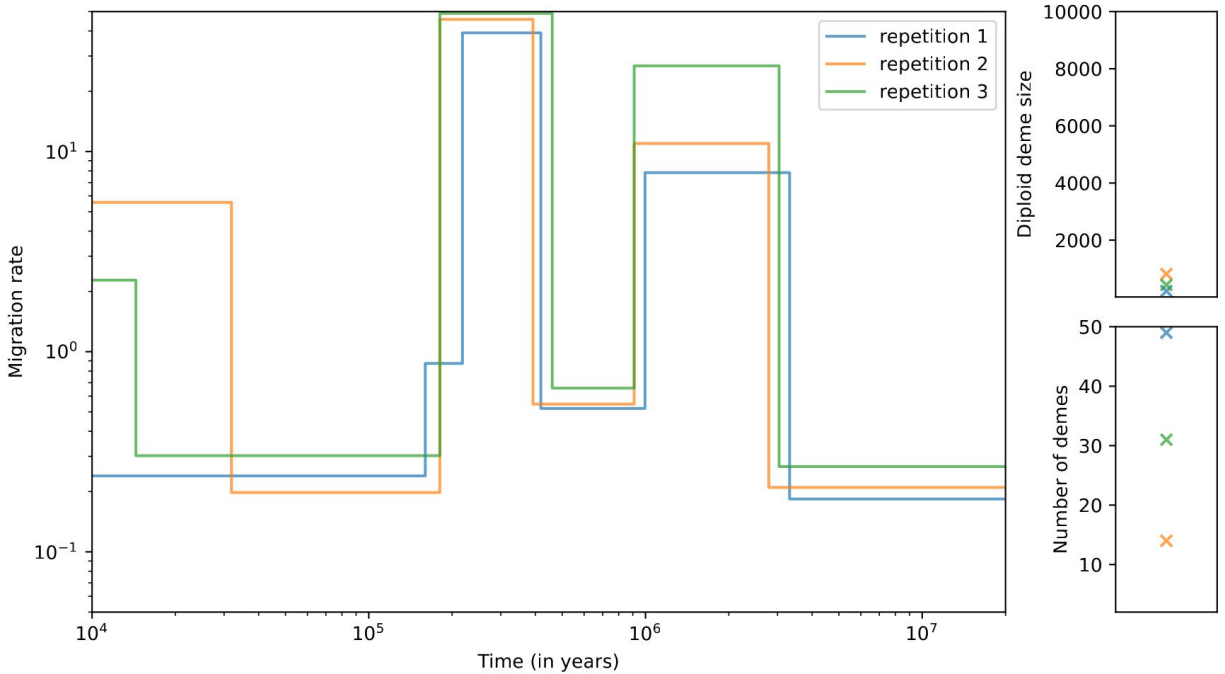
Distances between source and inferred IICR:
- 864436
- 992524
- 967878



SNIF in practice
Running SNIF on a PSMC file

- **basename_frompsmc_CG.pdf** ➜ shows the inferred demographic parameters

c = 6
w = 0.5
3 repetitions
30 rounds



SNIF in practice

Refining the parameter space

We advise to:

1. First identify the best inference parameters (in particular **c** and w) that fit your data

c = number of time components: the higher the better the fit, but the more parameters to inter

SNIF in practice
Refining the parameter space

We advise to:

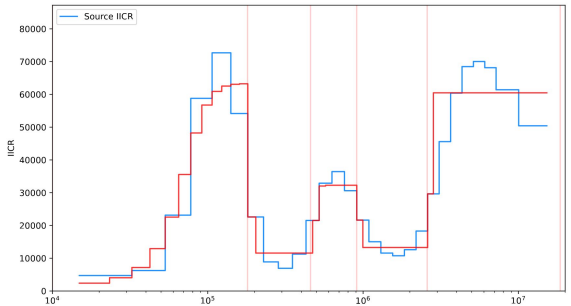
1. First identify the best inference parameters (in particular **c** and w) that fit your data

c = number of time components: the higher the better the fit, but the more parameters to inter

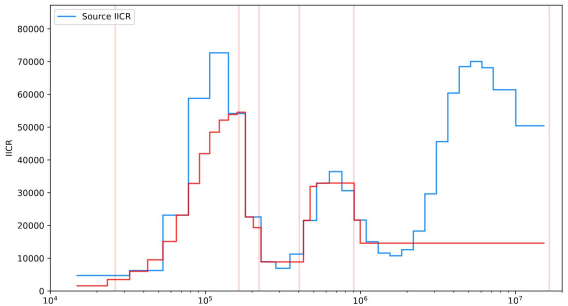
w = gives more or less weight to the ancient past in the fitting of the input PSMC or IICR:

- $w < 1$ gives more weight to the ancient past
- $w > 1$ gives more weight to the recent past

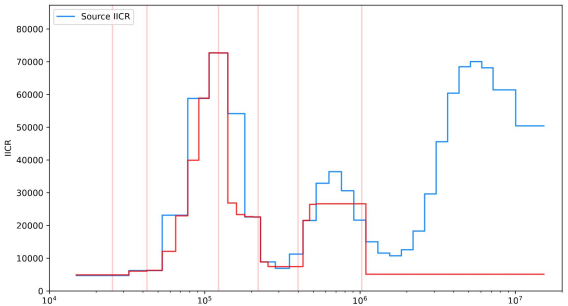
w = 0.5



w = 1



w = 2



SNIF in practice

Refining the parameter space

We advise to:

1. First identify the best inference parameters (in particular **c** and $\boldsymbol{w}$) that fit your data by using:
    a. “for” loops to test several inference parameter values
    b. a wide demographic parameter space
    c. a low number of repetitions (~2-3)

SNIF in practice

Refining the parameter space

We advise to:

1. First identify the best inference parameters (in particular **c** and $\boldsymbol{w}$) that fit your data by using:
    a. “for” loops to test several inference parameter values
    b. a wide demographic parameter space
    c. a low number of repetitions (~2-3)
2. Once the best inference parameters are identified:
    a. refine the demographic parameter space to allow quicker convergence
    b. use a higher number of repetitions (>10)

SNIF in practice
Refining the parameter space

1. Let us open ./workshop_files/param_files/param_Ptroglodytes_verus_Donald_psmc.py
2. Comment the lines 31, 33 and 35
3. Uncomment the lines 32, 34, 36.
4. Let us run SNIF again:
    a. make sure that the parameter file is located in snif-main
    b. open a terminal
    c. run `python3 param_Ptroglodytes_verus_Donald_psmc.py`

SNIF in practice

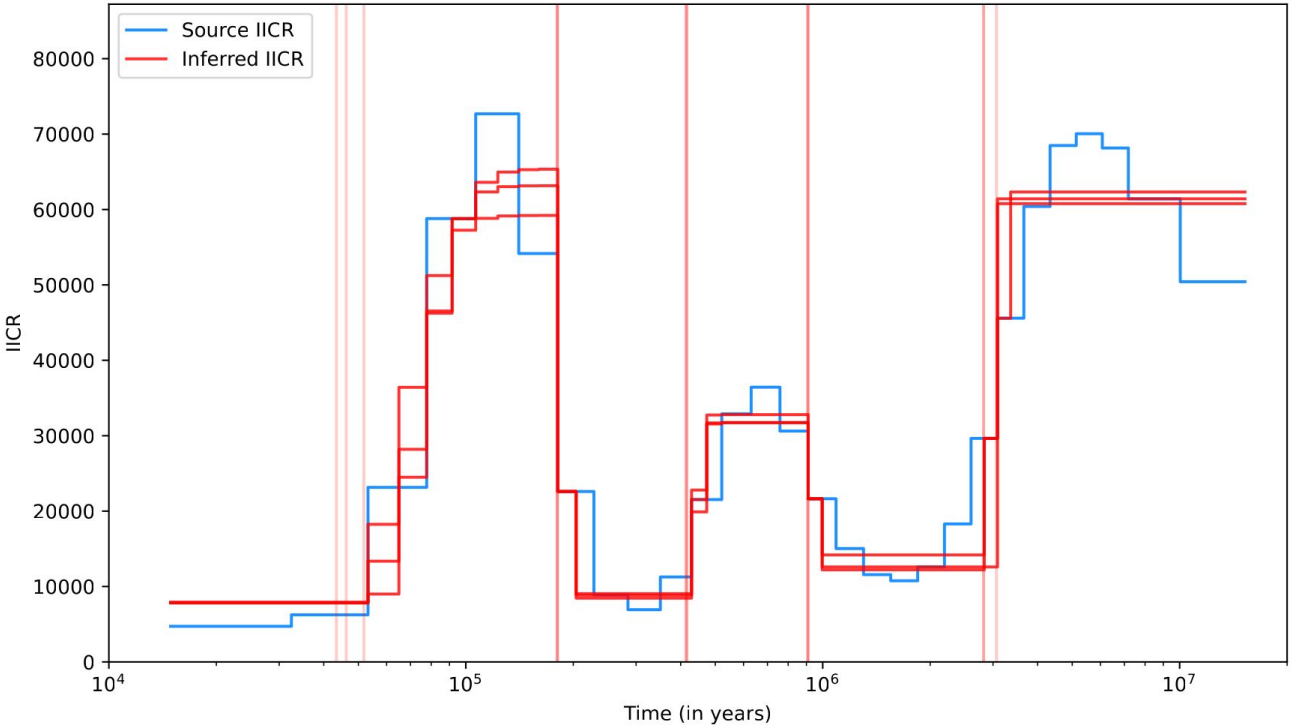
Refining the parameter space

Source PSMC and inferred IICR

c = 6
w = 0.5
3 repetitions
30 rounds

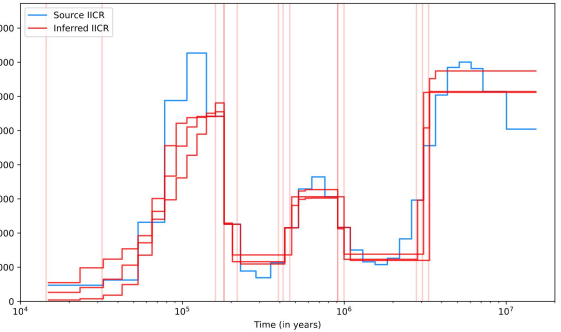
Distances between source and inferred IICR:

- 818843
- 806861
- 822110



SNIF in practice

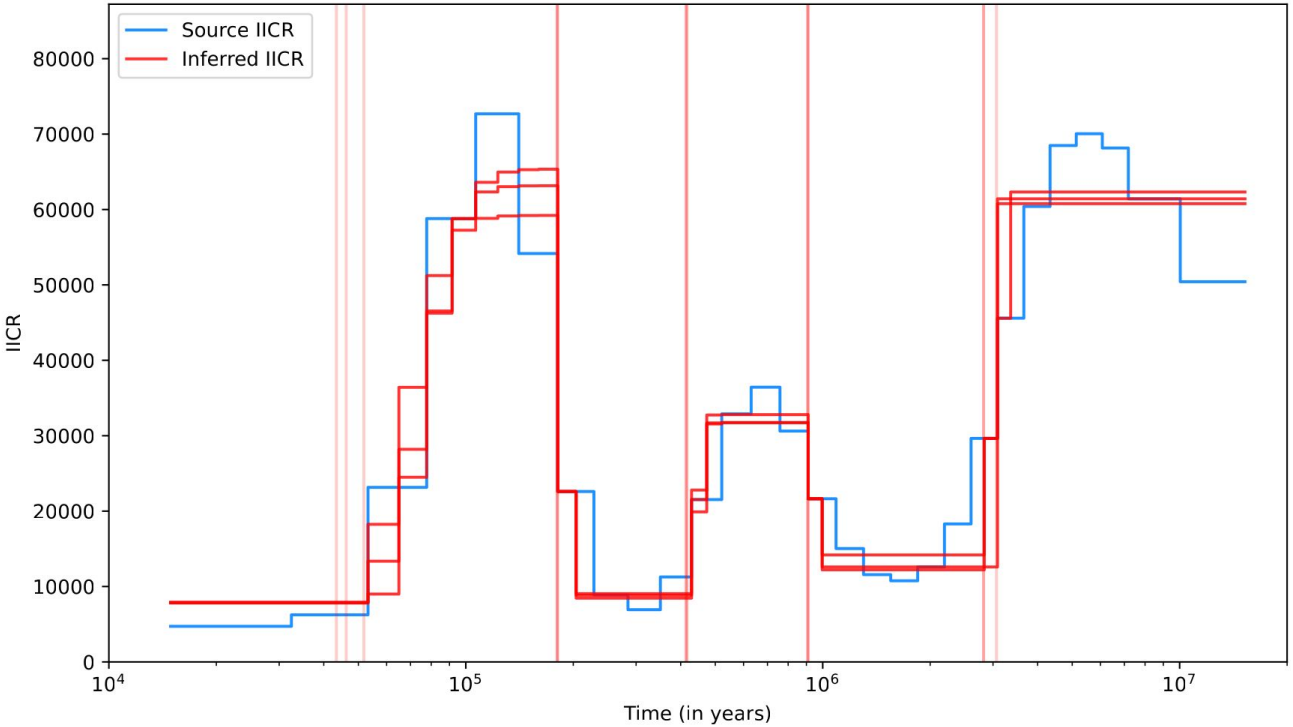
Refining the parameter space



Distances between source and inferred IICR:

- 864436
- 992524
- 967878

- 818843
- 806861
- 822110

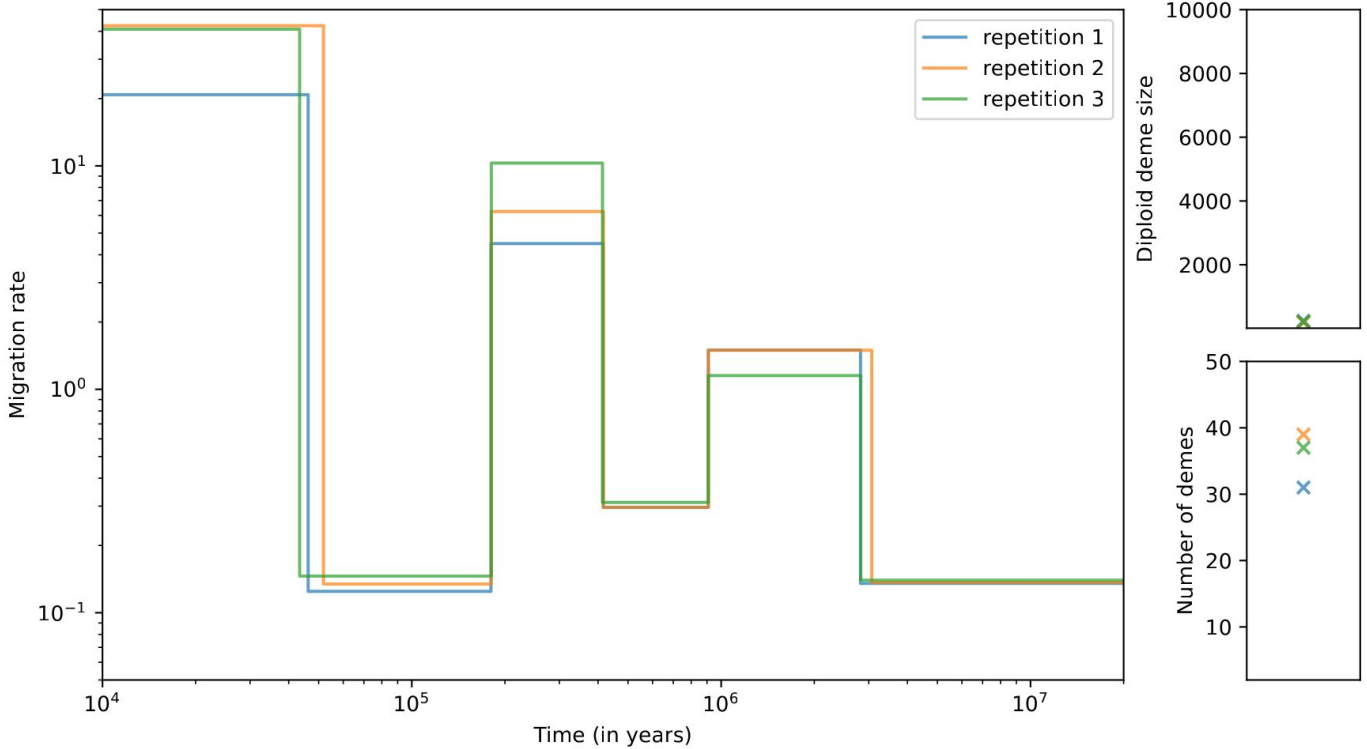


SNIF in practice

Refining the parameter space

Inferred demographic parameters

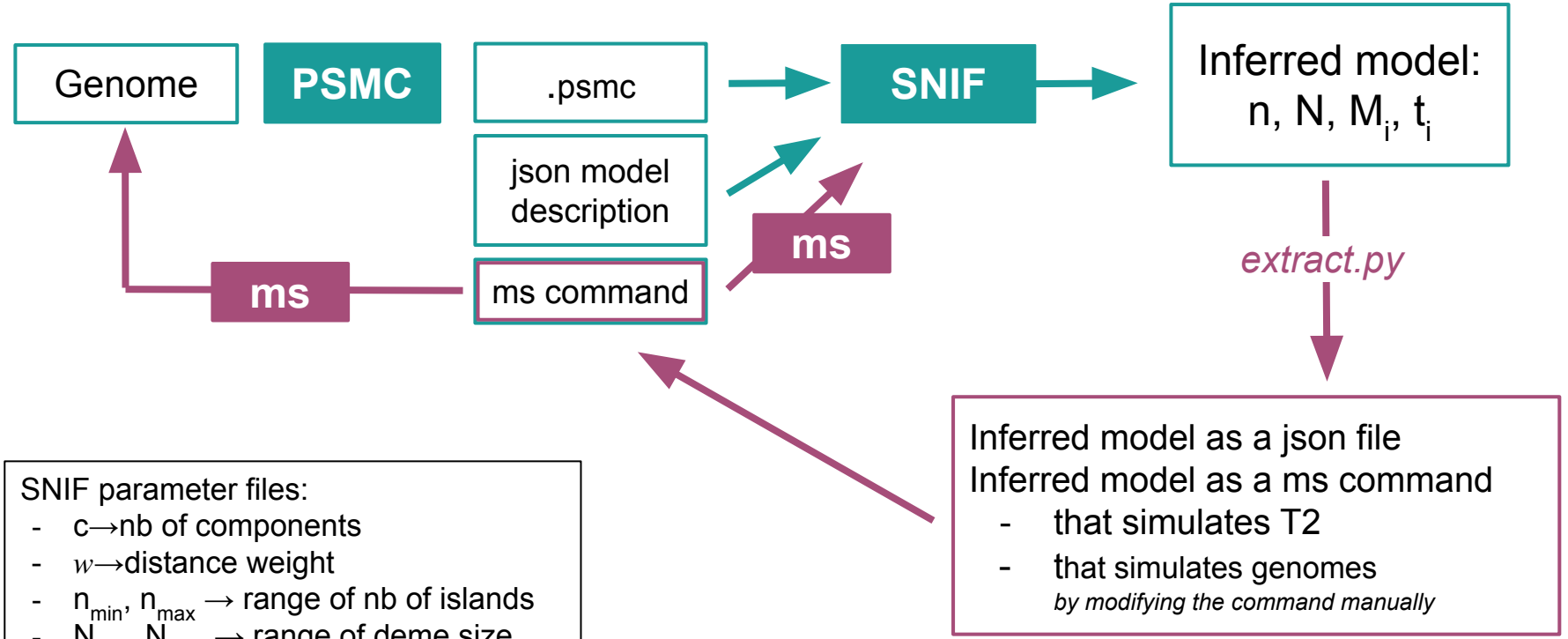
c = 6
w = 0.5
3 repetitions
30 rounds



SNIF in practice

Validating the inferred scenarios

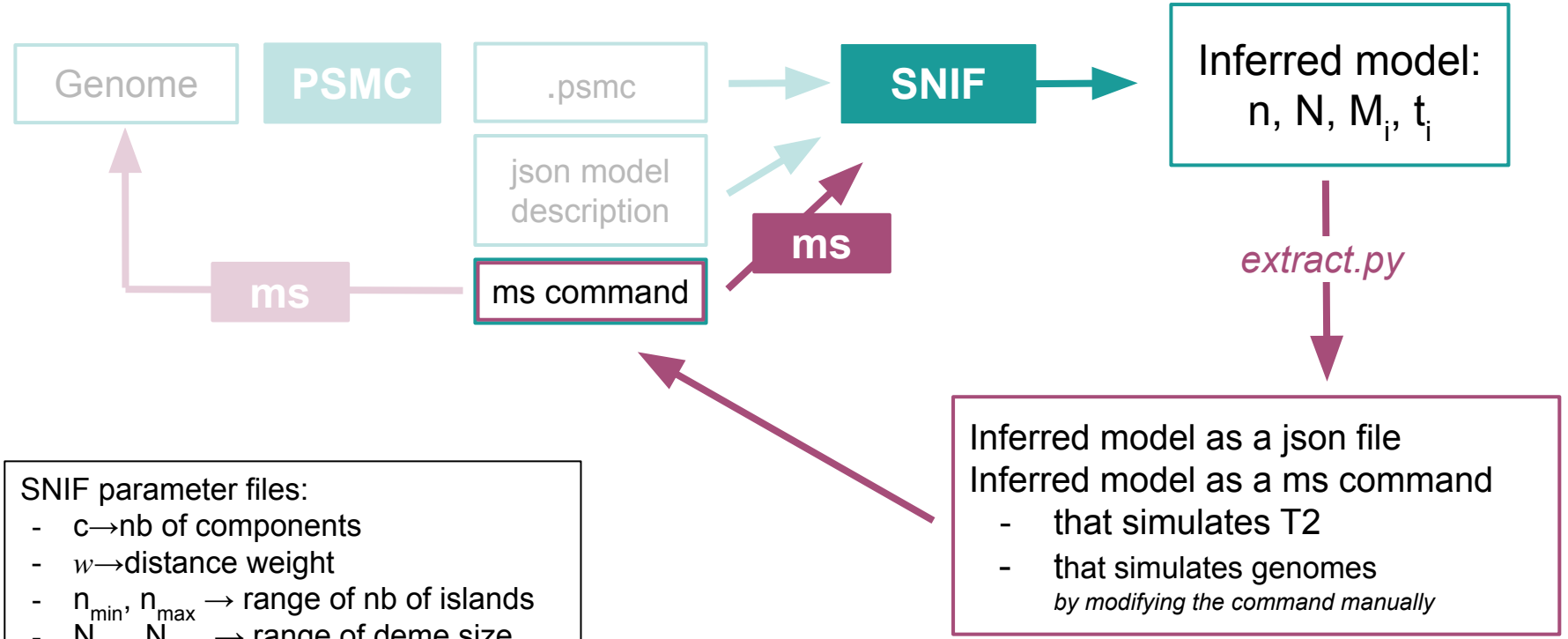
- curves: csv file
- inferred parameters: csv file
- settings: json file



SNIF in practice

Validating the inferred scenarios

- curves: csv file
- inferred parameters: csv file
- settings: json file



SNIF parameter files:

- c→nb of components
- w→distance weight
- n_{min}, n_{max} → range of nb of islands
- N_{min}, N_{max} → range of deme size
- …

SNIF in practice
Validating the inferred scenarios

i is the id number of the repetition you want to extract

We will now run SNIF on a scenario inferred in the first inference step. We will:

1. Extract an inferred scenario as a ms command using the script ./extract.py and the command:

```
python3 extract.py ./workshop_files/SNIF_results/basename -ms i
```

   You should now have a file in ./workshop_files/SNIF_results/ named basename_inferred-00i.ms. Copy and paste it to ./workshop_files/input_data/.

SNIF in practice
Validating the inferred scenarios

i is the id number of the repetition you want to extract

We will now run SNIF on a scenario inferred in the first inference step. We will:

1. Extract an inferred scenario as a ms command using the script ./extract.py and the command:

```
python3 extract.py ./workshop_files/SNIF_results/basename -ms i
```

   You should now have a file in ./workshop_files/SNIF_results/ named basename_inferred-00i.ms. Copy and paste it to ./workshop_files/input_data/.

2. Extract the same inferred scenario as a .json file using the same script ./extract.py and the command:

```
python3 extract.py ./workshop_files/SNIF_results/basename i
```

   You should now have a file in ./workshop_files/SNIF_results/ named basename_inferred-00i.json. Copy and paste it to ./workshop_files/input_data/.

SNIF in practice
Validating the inferred scenarios

3. Run SNIF

The input file is the extracted ms command that you placed in ./workshop_files/input_data/.
The parameter file we will use is: ./workshop_files/param_files/param_Ptroglodytes_verus_Donald_msT2.py

- Add the path to the input ms command (parameter “data_source”)
- Add the inferred diploid deme size (parameter “ms_reference_size”)
- Add the path to the .json file at the beginning of the script (variable “target_scenario”, line 5)

- Copy and paste the parameter file in snif-main
- Open a terminal in snif-main
- Run SNIF with: `python3 param_Ptroglodytes_verus_Donald_msT2.py`

```
(spyder-env) csteux@PP04018:~/Documents/Doctorat/SNIF/workshop_devocgen/snif-main$ python3 param_Ptroglodytes_verus_Donald_msT2.py

Welcome to SNIF -- an Inferential Framework for Demographic Structure
found 1 source file of type MSCommand
loading static library..
initializing static library..
D Runtime initialized succesfully

loading file 1 of 1: ./workshop_files/input_data/Pan_troglodytes_verus_Donald_inferred-model-004.ms
simulating T2 samples... done.
repetition 1 of 2
  round 1 of 30
```

SNIF in practice
Validating the inferred scenarios

You should have **6 files** created in ./workshop_files/SNIF_results/:

- **basename_frompsmc.csv** = inferred parameters and other inference statistics
- **basename_frompsmc_curves.csv** = the source IICR and the inferred IICR
- **basename_frompsmc_settings.json** = summary of the parameters used for the inference
- **basename_fromT2.out** = the simulated T2 used to create the source IICR

The plots were created using the script snif2plot.py, and therefore you should also have:

- **basename_frompsmc_IICR.pdf** ➜ shows the fit of the inferred IICR to the input PSMC curve
- **basename_frompsmc_CG.pdf** ➜ shows the inferred demographic parameters

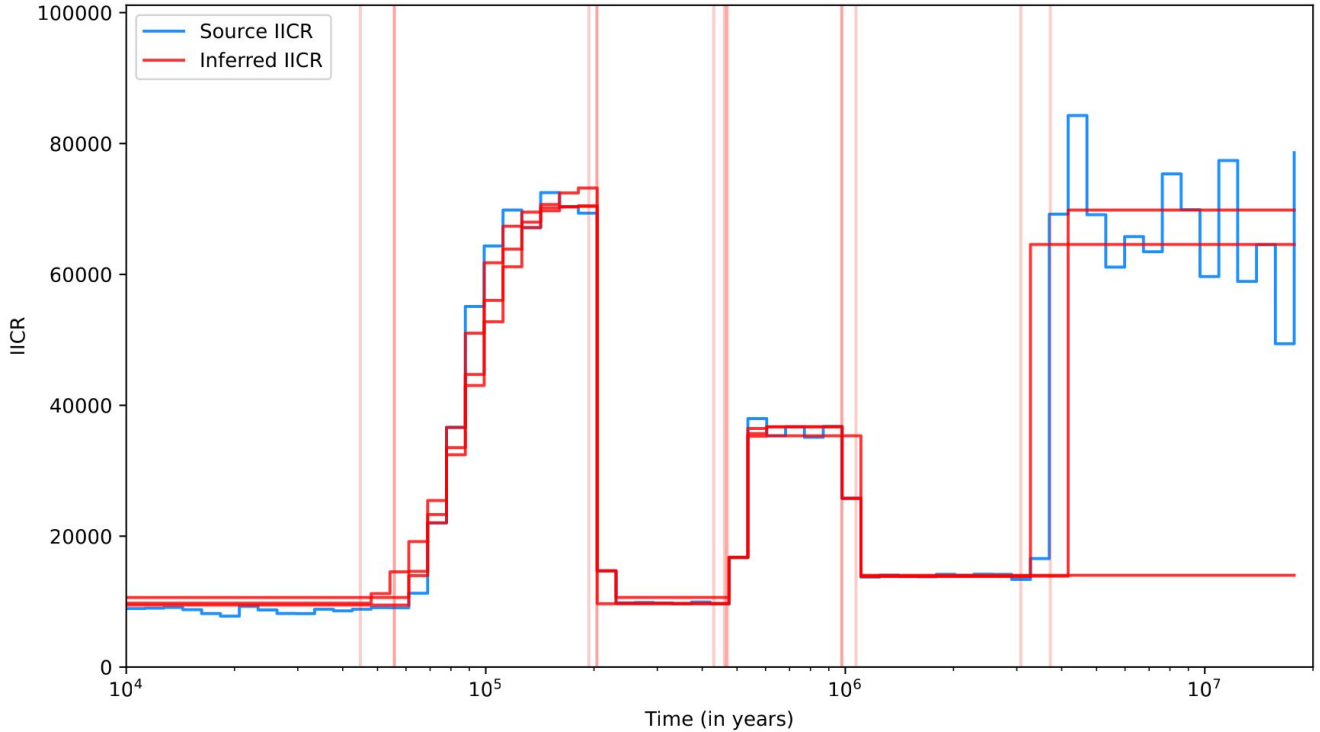
Note that if several repetitions were done with SNIF, all the inferred scenarios are in the same output files.

SNIF in practice

Validating the inferred scenarios

Inferred demographic parameters

c = 6
w = 0.5
3 repetitions
30 rounds

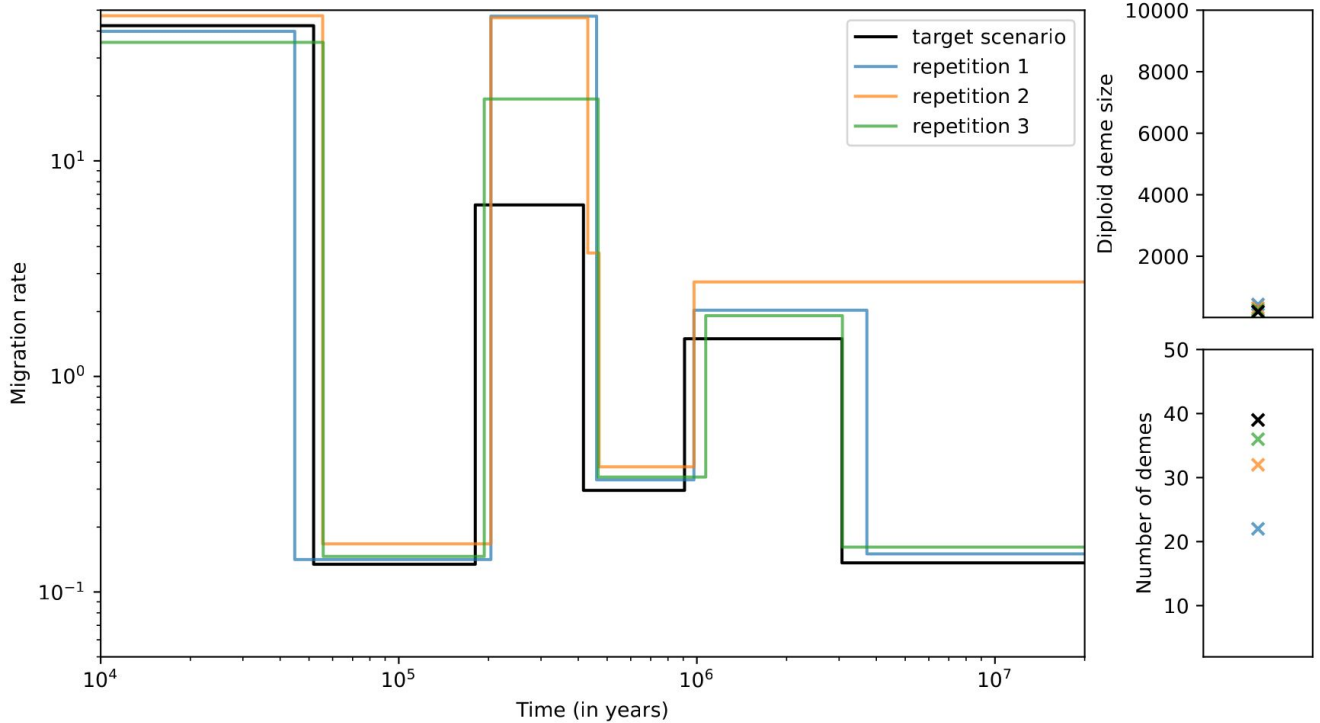


SNIF in practice

Validating the inferred scenarios

Inferred demographic parameters

c = 6
w = 0.5
3 repetitions
30 rounds



Getting started with SNIF

Running SNIF on a ms command that simulates genomes

Note that you can also give to SNIF as input data a ms command that simulates genomes. In that case, SNIF will run ms to simulate the genomes, run PSMC on the simulated genomes and do the inference.

You will have to **adapt the extracted ms command manually** so it simulates genomes instead of T2.

Use the option “-p 12” so the resolution of the genomic position is high enough.

An example of a ms command that simulates 10 chromosomes of 100 Mb can be found in ./workshop_files/input_data/Pan_troglodytes_verus_Donald_inferred-model-004_10x100Mb.ms

An example of a parameter file to run SNIF on a ms command that simulates genomic data can be found in ./workshop_files/param_files/param_Ptroglodytes_verus_Donald_ms10x100Mb.py.

Getting started with SNIF
Running SNIF on a json file

Note that you can also give to SNIF the parameters of a piecewise stationary n-island model in the .json format. In that case, SNIF will analytically derive the IICR from the demographic scenario and do the inference.

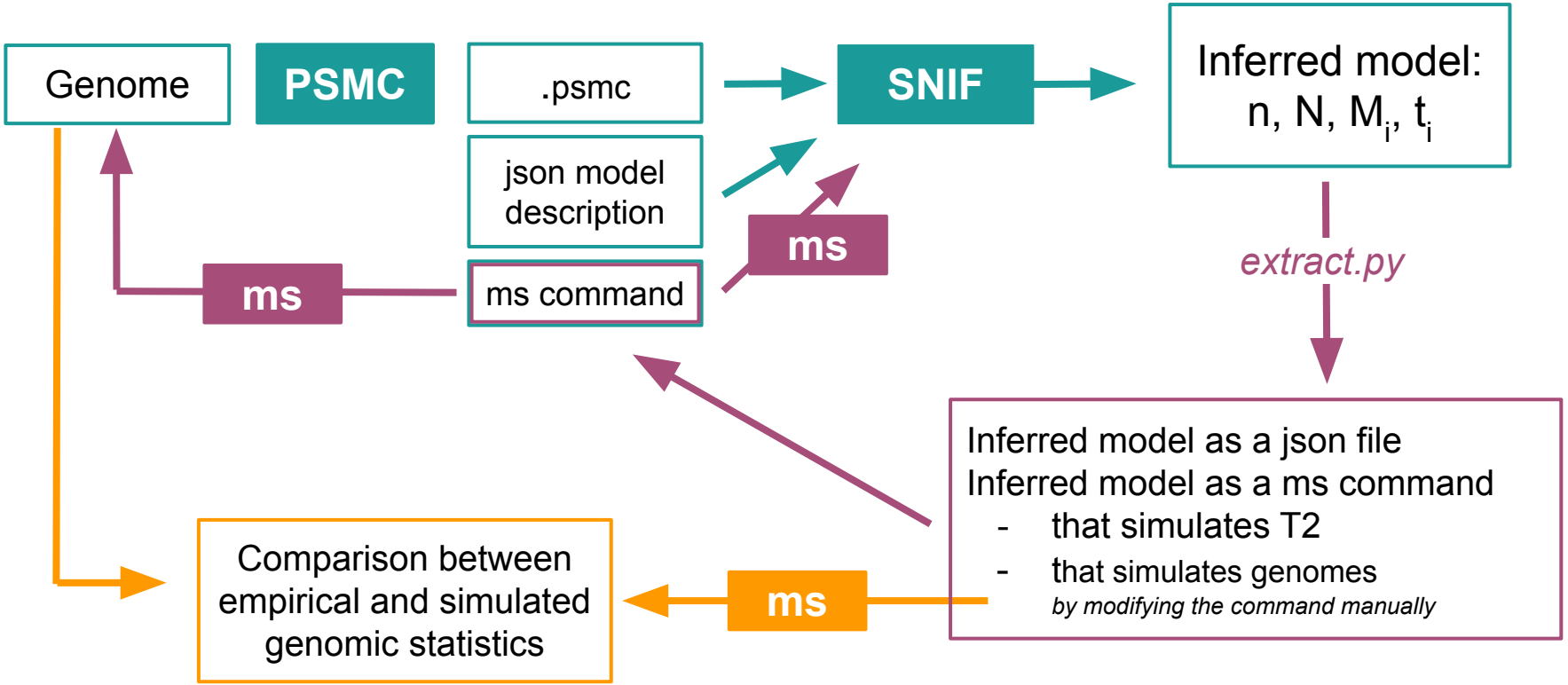
It is the same .json file that can be extracted using the script ./extract.py and used to plot a target scenario.

An example of a .json file can be found in
./workshop_files/input_data/Pan_troglodytes_verus_Donald_inferred-model-004.json

An example of a parameter file to run SNIF on a ms command that simulates genomic data can be found in ./workshop_files/param_files/param_Ptroglodytes_verus_Donald_json.py.

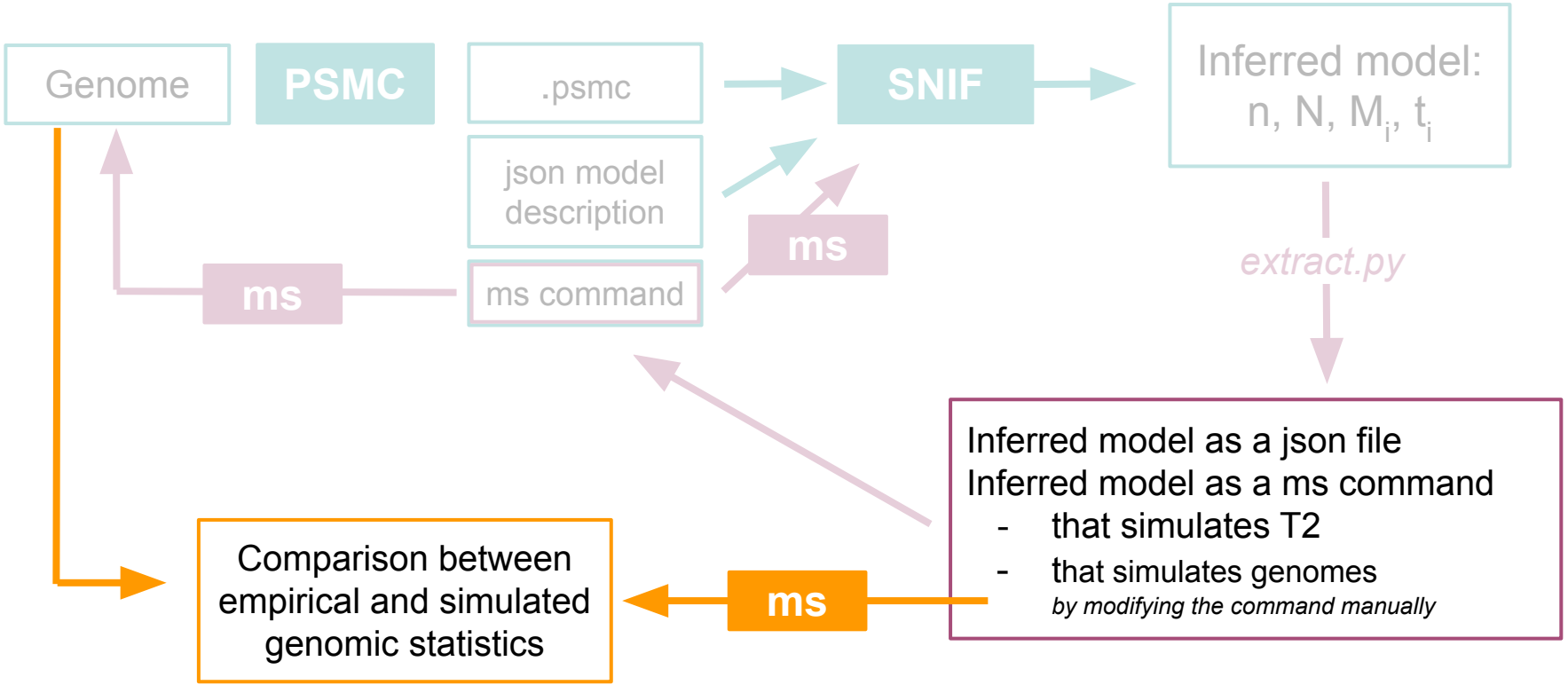
SNIF in practice
Further validation

- curves: csv file
- inferred parameters: csv file
- settings: json file



SNIF in practice
Further validation

- curves: csv file
- inferred parameters: csv file
- settings: json file



SNIF in practice
Further validation

Finally, it can be interesting to further validate the inferred scenarios by computing genomic statistics on simulated data: genetic diversity, genetic differentiation, …

```
(spyder-env) csteux@PP04018:~/Documents/Doctorat/SNIF/workshop_devocgen$ python3 compute_het_from_
ms.py -ms Pan_troglodytes_verus_Donald_inferred-model-004_100x10Mb.out
Computing individual heterozygosity.
0.0011  ( 0.00106 , 0.001187 )
```

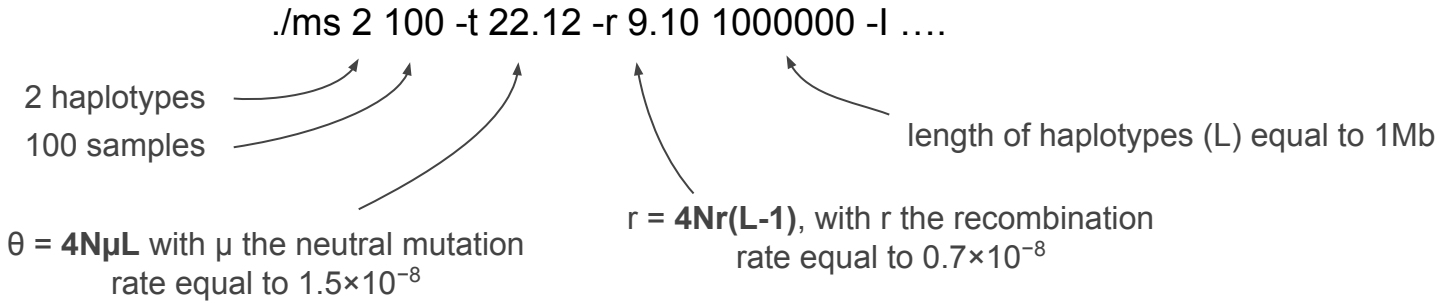
0.0011 / 1000 = 1.1
heterozygous site per kb

Pan_troglodytes_verus-9668_Bosco	991,297	0.58
Pan_troglodytes_verus-9730_Donald	1,646,466	0.96
Pan_troglodytes_verus-A956_Jimmie	1,006,068	0.58
Pan_troglodytes_verus-A991_Berta	797,536	0.46
Pan_troglodytes_verus-A992_Annie	915,718	0.53
Pan_troglodytes_verus-A993_Mike	889,689	0.52
Pan_troglodytes_verus-B005_SeppToni	1,079,046	0.63
Pan_troglodytes_verus-B006_Linda	1,070,960	0.62
Pan_troglodytes_verus-Clint	990,108	0.58
Pan_troglodytes_verus-N014_Cindy	1,069,613	0.62
Pan_troglodytes_verus-N016_Alice	1,024,124	0.60
Pan_troglodytes_verus-X00100_Koby	977,227	0.57

Empirical individual observed heterozygosity per kb. for Western chimpanzees. From de Manuel et al. 2016

SNIF in practice
Further validation

1. Modify the beginning of the extracted ms command to simulate genomic data as:



2. Place it in snif-main and open a terminal in snif-main
3. Simulate the genomic data using:

```
bash Pan_troglodytes_verus_Donald_inferred-model-004_100x1Mb.ms > genomic_data.out
```

4. Run: `python3 ../compute_het_from_ms --ms_out genomic_data.out`

General discussion on the concept of effective population size

Downloading SNIF and other necessary programs

SNIF
SNIF scripts can be downloaded from: https://github.com/arredondos/snif
Download the whole directory (green button “Code” and “Download ZIP”)
Unzip it and place it where you want SNIF to be.

ms (Hudson 2001)
If you are planning on using ms commands as input, you will need to install ms as well.
Download the scripts from http://home.uchicago.edu/~rhudson1/source/mksamples.html
then LINK and download the ms.tar.gz and the msdoc.pdf files.
Unzip it and then follow the instructions given in the section “Downloading and compilation” in the msdoc.pdf.
Then place the executable ms in the snif-main directory (copy and paste).

PSMC (Li and Durbin 2011)
If you are planning on using SNIF on ms commands that simulated genomic sequences, you will need to install PSMC as well.
Download the program from https://github.com/lh3/psmc and copy and paste the psmc executable in the snif-main directory.