Constraining tip dates to non-zero values using tsinfer/tsdate? Part II.

In [28]:
```
import tsinfer
import tskit
import tsdate
import numpy as np
import pandas as pd
import datetime as dt
import time
```

infer tree sequence

In [29]:
```
vdata = tsinfer.VariantData("chr1/chr1.vcz",
                            ancestral_state="variant_AA")
```

In [30]:
```
inferred_ts = tsinfer.infer(vdata)
print(f"Inferred a genetic genealogy for {inferred_ts.num_samples} (haploid) genomes")
```

```
Inferred a genetic genealogy for 29 (haploid) genomes
```

store tip dates in sampling_times array

In [31]:
```
metadata = pd.read_table("data/filtered_metadata.tsv")
```

In [32]:
```
# get sampling times
metadata['time'] = metadata['date'].str[:4].astype(float)
# set time as N years since most recent sample
metadata['time']  = max(metadata['time']) - metadata['time']
#metadata['time']  = 2025 - metadata['time']
```

In [33]:
```
# create sample:time dictionary
accession_to_time = dict(zip(metadata['experiment_accession'], metadata['time']))

# match tip names and sampling times to nodes names
individuals_list = {}

for s in inferred_ts.samples():
    individuals_list[s] = inferred_ts.individual(inferred_ts.node(s).individual).metadata['variant_data_sample_id']

sampling_times = np.full(inferred_ts.num_individuals, tskit.UNKNOWN_TIME)

for sample in inferred_ts.samples():
    id = individuals_list[sample]
    sampling_times[sample] = accession_to_time[id]
```

In [34]:
```
sampling_times
```

Out[34]:
```
array([17., 49., 35., 32.,  4., 38., 28., 29., 50., 43., 22., 44., 16.,
       30., 26., 52., 49., 30., 30., 33., 10., 10., 37., 33., 86., 48.,
        2.,  0.,  9.])
```

reinfer tree sequence with sampling_times array

In [35]:
```
vdata = tsinfer.VariantData("chr1/chr1.vcz",
                            ancestral_state="variant_AA",
                            individuals_time = sampling_times)
```

In [36]:
```
# inferred_ts = tsinfer.infer(vdata, # tip dates are stored in vdata
#                             time_units = "years")

# print(f"Inferred a genetic genealogy for {inferred_ts.num_samples} (haploid) genomes")
```

(1) generate ancestors

In [37]:
```
anc = tsinfer.generate_ancestors(vdata)
```

(2) match ancestors

In [38]:
```
anc_m = tsinfer.match_ancestors(vdata, anc, time_units = "years")
anc_m.tables.nodes.time
```

Out[38]:
```
array([1.11405836, 1.0397878 , 0.96551724, 0.93103448, 0.65517241,
       0.62068966, 0.55172414, 0.34482759, 0.31034483, 0.27586207,
       0.27586207, 0.20689655, 0.17241379, 0.17241379, 0.17241379,
       0.13793103, 0.13793103, 0.13793103, 0.10344828, 0.10344828,
       0.10344828, 0.10344828, 0.06896552, 0.06896552, 0.06896552,
       0.06896552, 0.06896552, 0.06896552])
```

(3) match samples [error]

In [39]:
```
anc_ts = tsinfer.match_samples(vdata, anc_m, force_sample_times=True)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[39], line 1
----> 1 anc_ts = tsinfer.match_samples(vdata, anc_m, force_sample_times=True)

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:1629, in match_samples(variant_data, ancestors_ts, recombination_rate, mismatch_ratio, path_compression, indexes, post_process, force_sample_times, num_threads, o
verlay_non_inference_sites, recombination, mismatch, precision, extended_checks, engine, progress_monitor, simplify, record_provenance, results)
   1622     sample_times = variant_data.individuals_time[:][individuals]
   1624     # Here we might want to re-order sample_indexes and sample_times
   1625     # so that any historical ones come first, any we bomb out early
   1626     # if they conflict but that would mean re-ordering the sample
   1627     # nodes in the final ts, and we sometimes assume they are in
   1628     # the same order as in the file
-> 1629 manager.match_samples(sample_indexes, sample_times, results)
   1630 ts = manager.finalise(overlay_non_inference_sites)
   1631 if post_process:

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:2766, in SampleMatcher.match_samples(self, sample_indexes, sample_times, results, slice_)
   2764 if slice_ is None:
   2765     slice_ = slice(0, len(sample_indexes))
-> 2766 return self._match_samples(sample_indexes[slice_], results)

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:2735, in SampleMatcher._match_samples(self, sample_indexes, results)
   2733 if np.any(times[node_id] > times[result.path.parent]):
   2734     p = result.path.parent[np.argmin(times[result.path.parent])]
-> 2735     raise ValueError(
   2736         f"Failed to put sample {j} (node {node_id}) at time "
   2737         f"{times[node_id]} as it has a younger parent (node {p})."
   2738     )
   2739 builder.add_path(
   2740     result.node,
   2741     result.path.left,
   (...)
   2744     compress=self.path_compression,
   2745 )
   2746 builder.add_mutations(
   2747     result.node,
   2748     result.mutations_site,
   2749     result.mutations_derived_state,
   2750 )

ValueError: Failed to put sample 0 (node 28) at time 17.0 as it has a younger parent (node 26).
```

With "force_sample_times = True", throws "ValueError: Failed to put sample 0 (node 28) at time 17.0 as it has a younger parent (node 26)." It seems that due to the uncalibrated times given by tsinfer, my real tip ages are always much older than any internal nodes.

In [40]:
```
# Now match_samples without forcing sample times
anc_ts = tsinfer.match_samples(vdata, anc_m) # ,force_sample_times=True)
anc_ts.tables.nodes.time
```

Out[40]:
```
array([0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.        ,
       0.        , 0.        , 0.        , 0.        , 0.06896552,
       0.06896552, 0.06896552, 0.06896552, 0.06896552, 0.06896552,
       0.10344828, 0.10344828, 0.10344828, 0.10344828, 0.10344828,
       0.13793103, 0.13793103, 0.13793103, 0.17241379, 0.17241379,
       0.17241379, 0.20689655, 0.27586207, 0.27586207, 0.31034483,
       0.34482759, 0.55172414, 0.62068966, 0.65517241, 0.93103448,
       0.96551724, 1.0397878 ])
```

In [41]:
```
inferred_ts = tsinfer.match_samples(vdata, anc_ts, force_sample_times=True)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Cell In[41], line 1
----> 1 inferred_ts = tsinfer.match_samples(vdata, anc_ts, force_sample_times=True)

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:1599, in match_samples(variant_data, ancestors_ts, recombination_rate, mismatch_ratio, path_compression, indexes, post_process, force_sample_times, num_threads, o
verlay_non_inference_sites, recombination, mismatch, precision, extended_checks, engine, progress_monitor, simplify, record_provenance, results)
   1597 variant_data._check_finalised()
   1598 progress_monitor = _get_progress_monitor(progress_monitor, match_samples=True)
-> 1599 manager = SampleMatcher(
   1600     variant_data,
   1601     ancestors_ts,
   1602     recombination_rate=recombination_rate,
   1603     mismatch_ratio=mismatch_ratio,
   1604     recombination=recombination,
   1605     mismatch=mismatch,
   1606     path_compression=path_compression,
   1607     num_threads=num_threads,
   1608     precision=precision,
   1609     extended_checks=extended_checks,
   1610     engine=engine,
   1611     progress_monitor=progress_monitor,
   1612 )
   1613 sample_indexes = check_sample_indexes(variant_data, indexes)
   1614 sample_times = np.zeros(
   1615     len(sample_indexes), dtype=variant_data.individuals_time.dtype
   1616 )

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:2668, in SampleMatcher.__init__(self, variant_data, ancestors_ts, **kwargs)
   2664 self.ancestors_ts_tables = ancestors_ts.dump_tables()
   2665 super().__init__(
   2666     variant_data, self.ancestors_ts_tables.sites.position, **kwargs
   2667 )
-> 2668 self.restore_tree_sequence_builder()
   2669 # Map from input sample indexes (IDs in the SampleData file) to the
   2670 # node ID in the tree sequence.
   2671 self.sample_id_map = {}

File ~/tsinfer-env/lib/python3.11/site-packages/tsinfer/inference.py:2341, in Matcher.restore_tree_sequence_builder(self)
   2336     raise ValueError(
   2337         "Ancestors tree sequence not compatible: sequence length is different to"
   2338         " sample data file."
   2339     )
   2340 if np.any(tables.nodes.time <= 0):
-> 2341     raise ValueError("All nodes must have time > 0")
   2343 edges = tables.edges
   2344 # Get the indexes into the position array.

ValueError: All nodes must have time > 0
```

run tsdate

In [42]:
```
# somehow this is still running despite ValueError from match_samples

simplified_ts = tsdate.preprocess_ts(
    inferred_ts,
    erase_flanks=True
)

dated_ts, fit = tsdate.date(simplified_ts,
                            mutation_rate=6.1e-7,
                            time_units="years",
                            return_fit=True,
                            match_segregating_sites = True,
                            #constr_iterations=100 # no effect on tip dates
                            )
```

In [47]:
```
dated_ts.nodes_time
```

Out[47]:
```
array([  0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   0.        ,   0.        ,   0.        ,
         0.        ,   8.1101485 ,   1.9017355 ,   9.54585197,
         0.92320977,   1.5683697 ,   5.42048973,   2.1821495 ,
         3.3631714 ,   3.84439326,   6.10075609,   2.21114753,
        17.71661843,   7.72050752,   5.87168088,  11.11856677,
        10.35199326,   9.95833546,  13.01286384,  11.0493302 ,
        13.87701521,  14.74093075,  16.83053289,  17.89812799,
```

```
 19.76832641,  21.66212157,  22.93135039,  24.58521966,
807.29412396,  10.66932169])
```