Adaptive Introgression Generating Function Functions

Here we model adaptive introgression after secondary contact ussing the model of Setter et al 2020. For analysis, we parameterize the generating function as a sequence of demographic events backward in time, measuring the time interval between successive vents. However, for many applications, we often assume that the species split at some fixed time Ts (s for split) in the past and that adaptive introgression occurs at some time 0 <= Ta <= Ts. Measuring the inter-event times, the first event is adaptive introgression ocurring after a time interval Ta. The second event is the subsequent split occurring after a second interval of length Ts-Ta.

Neutral Coalescent

Starlike Approximation

Instant Yule Approximation

Functions for Data

Results

n=2

AI. starlike only. gf function for n = 2 and expected time to MRCA

In[ ]:=
```
gfDeltaEpsilon = GFaiStar[α, ω, {{1}, {1}}, δa, δs];
gfDeltaEpsilon
gfTaEpsilon = InverseLaplaceTransform [gfDeltaEpsilon / δa, δa, Ta];
gfTaTs = InverseLaplaceTransform [gfTaEpsilon / δs, δs, Ts];
SubSimplifyPeeRulesv2 [gfTaTs , 2]
```

Out[ ]=
$$\frac{1 + \delta a\, P[0,\ 2] + \frac{\delta a\, \delta s\, P[1,2]}{(1+2\, \omega[\{1\}])\, (\delta s+2\, \omega[\{1\}])} + \frac{\delta a\, P[2,2]\left(1+\frac{\delta s}{1+2\, \omega[\{1\}]}\right)}{1+\delta s+2\, \omega[\{1\}]}}{1 + \delta a\, P[0,\ 2] + \delta a\, P[1,\ 2] + \delta a\, P[2,\ 2] + 2\, \omega[\{1\}]}$$

Out[ ]=
$$\frac{e^{Ta\, (-1-2\, \omega[\{1\}])-2\, Ts\, \omega[\{1\}]}\, P[1,\ 2]}{1 + 2\, \omega[\{1\}]} + \frac{1 - e^{Ta\, (-1-2\, \omega[\{1\}])}\, P[1,\ 2] + 2\, e^{Ta\, (-1-2\, \omega[\{1\}])}\, P[0,\ 2] \times \omega[\{1\}]}{1 + 2\, \omega[\{1\}]}$$

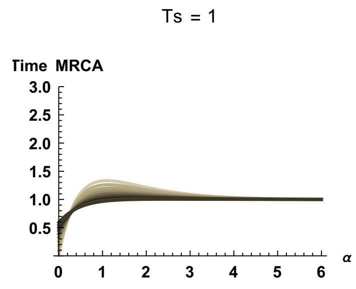
In[ ]:=
```
TmrcaAI[α_ , Ta_ , Ts_] = 1 / 2 * (-1 * D[gfTaTs , ω[{1}]] /. ω[_] → 0 /. P → PknToAlpha);
Print["Expected Total Singleton Branch Length"];
SubSimplifyPeeRulesv2 [-1 * D[gfTaTs , ω[{1}]] /. ω[_] → 0, 2] / 2 // FullSimplify
With[{TaIncrement = .25 , Ts = 4.0},
 Plot[
  Evaluate[Table[
    TmrcaAI[α, TaIncrement * i , Ts + TaIncrement * i], {i, 0, 8}]], {α, 0, 6}, PlotRange → All]
]
```

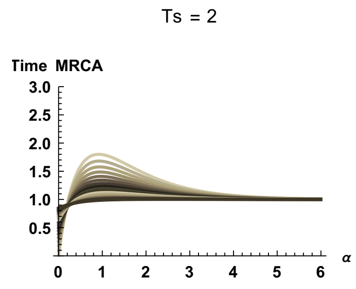
```
Expected Total Singleton Branch Length
```

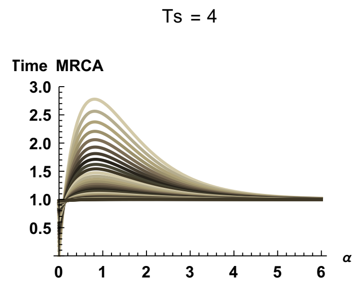
Out[ ]=
$$e^{-Ta}\left(e^{Ta} - P[0,\ 2] + Ts\, P[1,\ 2]\right)$$

We recover the ‘Volcano’ pattern in pairwise genetic diversity described in Setter et al 2020: the loss of diversity is observed only very near the sweep center, and this valley of diversity narrows as the divergence of the donor increases. In contrast, diversity is increased in the flanking regions by an amount proportional to the divergence of the donor. As the time since the sweep increases, diversity recovers to background levels.

Out[ ]=







Marginal Distributions for n = 2

In[ ]:=
```
gfDeltaEpsilon = GFaiStar[α, ω, {{1}, {1}}, δa, δs];
selGFList = gfDeltaEpsilon // Expand; selGFList[[0]] = List;
selGFList
```

Out[ ]=

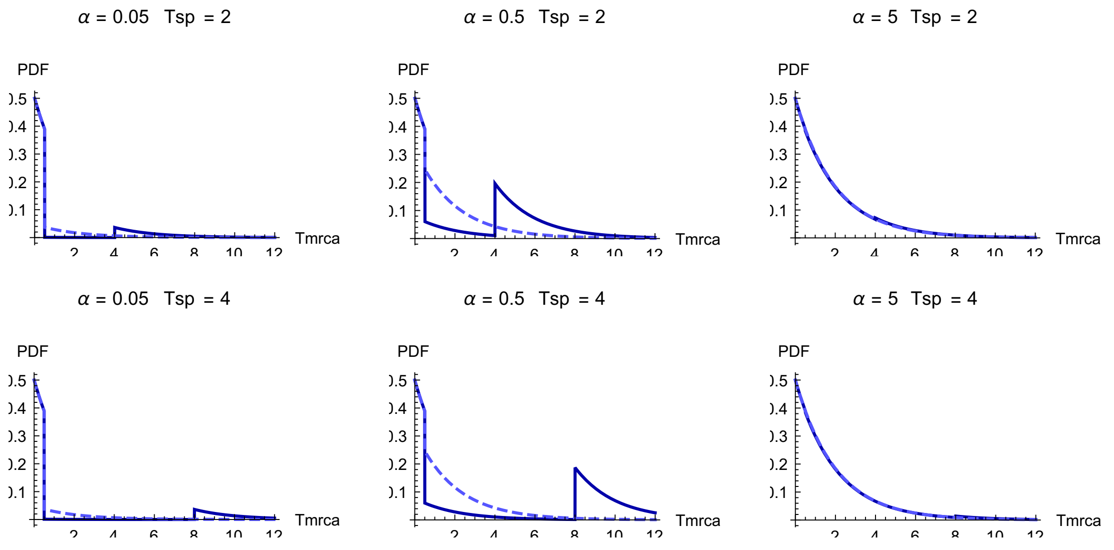
$$\left\{\frac{1}{1+\delta a\,P[0,2]+\delta a\,P[1,2]+\delta a\,P[2,2]+2\,\omega[\{1\}]},\ \frac{\delta a\,P[0,2]}{1+\delta a\,P[0,2]+\delta a\,P[1,2]+\delta a\,P[2,2]+2\,\omega[\{1\}]},\right.$$
$$\frac{\delta a\,\delta s\,P[1,2]}{(1+2\,\omega[\{1\}])\,(\delta s+2\,\omega[\{1\}])\,(1+\delta a\,P[0,2]+\delta a\,P[1,2]+\delta a\,P[2,2]+2\,\omega[\{1\}])},$$
$$\frac{\delta a\,P[2,2]}{(1+\delta s+2\,\omega[\{1\}])\,(1+\delta a\,P[0,2]+\delta a\,P[1,2]+\delta a\,P[2,2]+2\,\omega[\{1\}])},$$
$$\left.\frac{\delta a\,\delta s\,P[2,2]}{(1+2\,\omega[\{1\}])\,(1+\delta s+2\,\omega[\{1\}])\,(1+\delta a\,P[0,2]+\delta a\,P[1,2]+\delta a\,P[2,2]+2\,\omega[\{1\}])}\right\}$$

In[ ]:=
```
smList = FullSimplify[SubSimplifyPeeRulesv2[#, 2]] &/@
   MakeMeanList[selGFList, 2, ω, t, δa, Ta, δs, Ts];
spList = FullSimplify[SubSimplifyPeeRulesv2[#, 2]] &/@
   MakePDFList[selGFList, 2, ω, t, δa, Ta, δs, Ts];
scList = FullSimplify[SubSimplifyPeeRulesv2[#, 2]] &/@
   MakeCDFList[selGFList, 2, ω, t, δa, Ta, δs, Ts];
```

In[ ]:=
```
fT1[t_, Ta_, Ts_, α_] = spList[[1]] /. P → PknToAlpha;
FT1[t_, Ta_, Ts_, α_] = scList[[1]] /. P → PknToAlpha;
```

For the distribution of the time to the most recent common ancestor (solid lines, below), the selective sweep generates a burst of coalescent (and thereby a point-mass) at t=Ta in the same way as it does for the classic hard sweep case (dashed lines). In the adaptive introgression scenario, however, the lineages may be isolated from each other prior to the sweep and coalescence occurs only in the common ancestor population, resulting in a shift of the coalescent time farther past-ward that is proportional to the divergence time of the donor population, Ts.

Out[ ]=



In[ ]:=
```
GetJumps[spList, scList, t]
```

Out[ ]=
$$\{\{1, \{\text{DiracDelta}[t-2\,\text{Ta}], \text{HeavisideTheta}[t-2(\text{Ta}+\text{Ts})], \text{HeavisideTheta}[t-2\,\text{Ta}]\}, \{\{t-2\,\text{Ta}, e^{-\text{Ta}}\,P[0,2]\}\}\}\}$$

Distinguishing Ta from Ts

From the figure of expected tmrca above, it kinda looks like an old sweep from a highly divergent donor is indistinguishable from a recent sweep from a less-diverged donor. This is true, to an extent. Here, we use Td and Te for ease, representing the time to the sweep and the time between the sweep and the split respectively. If the sweep comes from a donor *y* times more divergent, then we obtain the same Tmrca if the time since the sweep with divergence value *x* where x is given below. We see that the two are distinguishable from the pattern of diversity >>along the genome <<. Note however, that if we ignore the α terms within the scaling factor (green line in figure below) we can get a reasonably accurate fit. This supports that pairwise diversity data alone is insufficient to infer all three parameters of the adaptive introgression sweep. however, you can already see the difference in the effects of Td and Te if you look at the marginal distribution.

In[ ]:=
```
Clear[x]
```

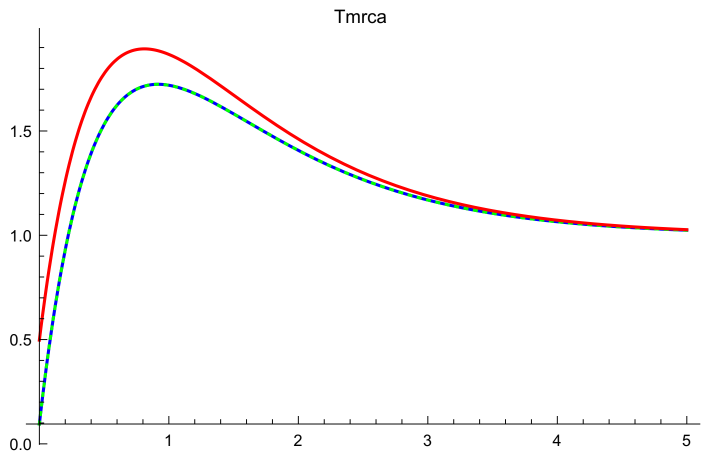
In[ ]:=
```
Solve[TmrcaAI[α, Td, Te] == TmrcaAI[α, x, Te * y], x]
```

Out[ ]=
$$\left\{\left\{x \to \text{Log}\left[\frac{e^{\text{Td}}\left(-1-2\,\text{Te}\,y+2\,e^{\alpha}\,\text{Te}\,y\right)}{-1-2\,\text{Te}+2\,e^{\alpha}\,\text{Te}}\right]\right\}\right\}$$

In[ ]:=
$$\text{Assuming}\left[\text{Td} > 0\ \&\&\ \text{Te} > 0, \text{FullSimplify}\left[\text{Log}\left[\frac{e^{\text{Td}}\left(-1-2\,\text{Te}\,y+2\,e^{\alpha}\,\text{Te}\,y\right)}{-1-2\,\text{Te}+2\,e^{\alpha}\,\text{Te}}\right]\right]\right]$$

Out[ ]=
$$\text{Td}+\text{Log}\left[\frac{-1+y}{-1+2\left(-1+e^{\alpha}\right)\text{Te}}+y\right]$$

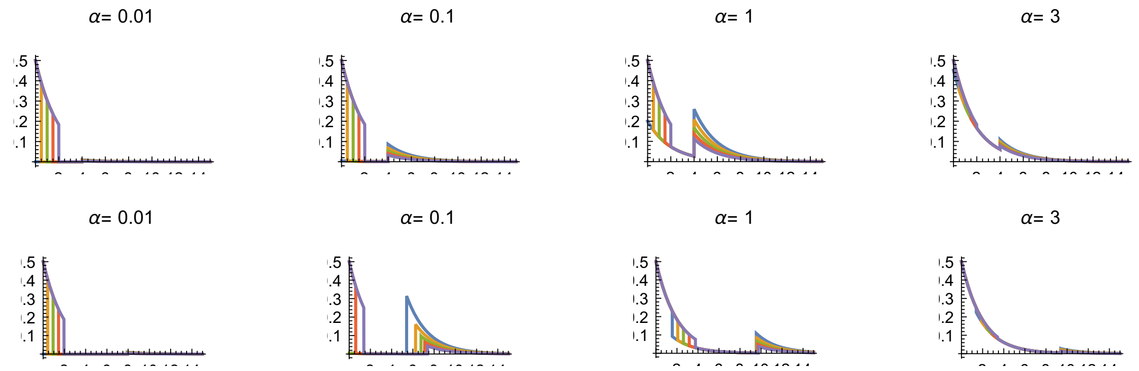
We see that by increasing the divergence between the donor and recipient populations, we can recover the same diversity by increasing the time since the sweep by the above factor. This does indeed depend on α to get correct answer (green dotted line matches blue line below). However, we see that ignoring terms of alpha to rescale Td-> $\mathrm{Log}\left[\frac{e^{Td}\,(-1-2\,Te\,y)}{-1-2\,Te}\right]$ yields a decnet approximation reasonable values of Td, Te, and y, although approximation is less accurate near the sweep center. This could prove problematic in trying to co-estimate all three values using measures of genetic diversity alone, particularly in that adaptive introgression leaves a strong and lasting signal in the periphery of the sweep region rather than at the center.

Out[ ]=


By allowing Td to rescale as a function of alpha, we correctly estimate the Tmrca (dashed green lines match the blue line in the plot above). However, even this rescaling does not correctly recover the underlying distribution of Tmrca along the genome. The distribution of Tmrca is well approximated with the rescaling factory both very near the sweep target and relatively far from the sweep target (α = 0.01 and α = 3, below). However, in the region flanking the sweep target, we can clearly distinguish the effects of the time since the sweep Td=Ta and the divergence of the donor at the time of introgression Te = Ts - Ta.

The reason is simple. As we saw in the marginal distribution and point mass (above), the burst of coalesence depends only on Ta, while the split time Ts determines how far pastward the probability mass is shifted relative to the classic sweep (Where Ts -> Ta).

Out[▫]=



n=4: marginals with Yule and Star Compared

For the analysis, we parameterize by the inter - event times Td = Ta and Te = Ts - Ta. If needed, we later substitute to recover as functions of Ts and Ta.

starlike: AI vs Classic

In[▫]:=
```
gfDeltaEpsilonStar = GFaiStar[α, ω, {{1}, {1}, {1}, {1}}, δ, ϵ] // Expand;
gfDeltaEpsilonStar [[0]] = List;
gfDeltaEpsilonStar =
  gfDeltaEpsilonStar //. subAllPeeToOne [4] //. subAllPeeToEpsilon [4];
```

The expressions become complicated, so we obtain the marginal iTon branch type means, pdfs, and cdfs below without printing them.

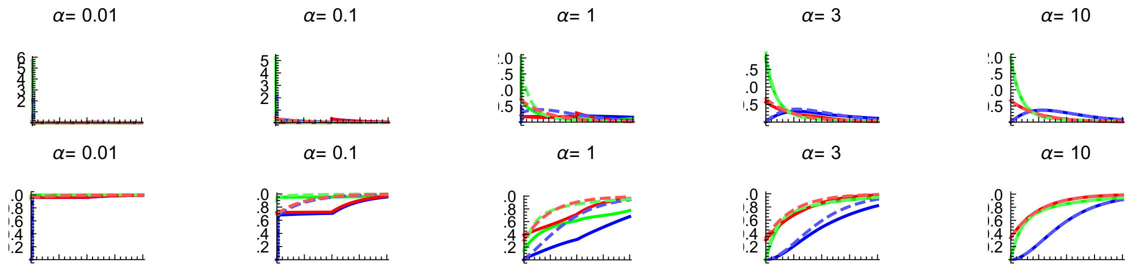
In[▫]:=
```
meansList = MakeMeanList[gfDeltaEpsilonStar , 4, ω, t, δ, Td, ϵ, Te];
pdfsList = MakePDFList[gfDeltaEpsilonStar , 4, ω, t, δ, Td, ϵ, Te];
cdfsList = MakeCDFList[gfDeltaEpsilonStar , 4, ω, t, δ, Td, ϵ, Te];
```

In[▫]:=
```
funMeans = Function[{α, Td, Te}, Evaluate[# /. P → PknToAlpha]] & /@ meansList;
funPdfs = Function[{t, α, Td, Te}, Evaluate[# /. P → PknToAlpha]] & /@ pdfsList;
funCdfs = Function[{t, α, Td, Te}, Evaluate[# /. P → PknToAlpha]] & /@ cdfsList;
```

Very near and very far from the sweep center, the marginal branch length distributions for the adaptive introgression case (solid lines: 1-Ton = blue, 2-Ton = green, 3-Ton = red) are similar to the classic hard sweep case (dashed lines). However, at intermediate distances from the sweep center, the coalescence times for the AI model are shifted pastward relative to those of the classic sweep.

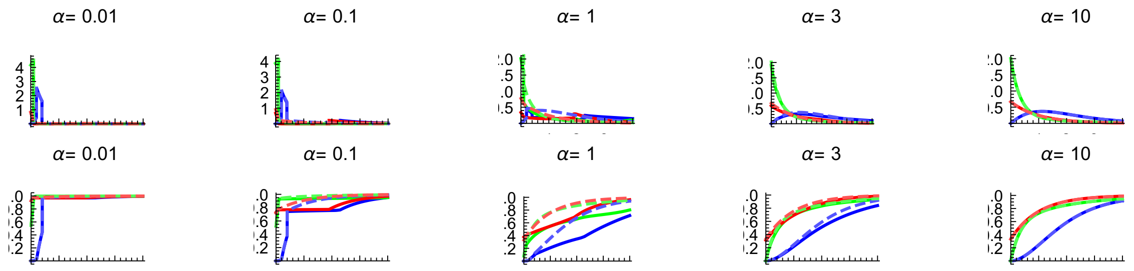
```
Ta = 0.01 , Ts = 2
```

Out[ ]=



```
Ta = 0.1 , Ts = 2
```

Out[ ]=



For singleton and doubleton lineages, a point-mass can only be generated by selection. Singletons can all coalesce in a single burst if the first event is the selective sweep and none of the lineages escape. Similarly, a point mass at t=0 results from a multiple merger of only 3 inidividuals. The size of these point masses depends only on Ta and the distance from the sweep target. However, for tripleton lineages, the size of the pointmass at t=0 also depends on the time since the split Ts, and this is due to the structure imposed on the coalescent by the strict isolation model. For example, suppose the sweep is the first event, and three out of four lineages escape. If the subsequent time to the population split is large, we expect all three of the escaped lineages to coalesce before coalescing with the donor lineage. In other words, for this sequence of events, the probability of observing tripleton-type branches is higher than under the classic sweep model due to the divergence time.

```
In[ ]:= GetJumps[pdfsList, cdfsList, t] /. Td → Ta /. Te → Ts - Ta // FullSimplify
Out[ ]= {{1, {HeavisideTheta[t - 4 Ta], HeavisideTheta[t - 2 Ta], HeavisideTheta[t - Ta],
      DiracDelta[t - 4 Ta], HeavisideTheta[t - 3 Ta - Ts], HeavisideTheta[t - Ta - Ts],
      HeavisideTheta[t - Ts], HeavisideTheta[t - 2 (Ta + Ts)], HeavisideTheta[t - 2 Ts],
      HeavisideTheta[t - 4 Ts], HeavisideTheta[t - 4 Ts]}, {{t - 4 Ta, e^(-6 Ta) P[0, 4]}}},
  {2, {HeavisideTheta[t - Ta], HeavisideTheta[t - 2 Ta], DiracDelta[t],
      HeavisideTheta[t + 2 Ta - 2 Ts], HeavisideTheta[t + Ta - 2 Ts], HeavisideTheta[t - 2 Ts],
      HeavisideTheta[t + Ta - Ts], HeavisideTheta[t - Ts]}, {{t, e^(-6 Ta) (P[0, 4] + P[1, 4])}}},
  {3, {DiracDelta[t], HeavisideTheta[t - Ta], HeavisideTheta[t + Ta - Ts],
      HeavisideTheta[t - Ts]}, {{t, 1/6 e^(-3 (Ta+Ts)) (-P[3, 4] + e^(-2 Ta+2 Ts) (-4 P[2, 4] + 3 P[3, 4] +
              2 e^(-Ta+Ts) (-1 + e^(6 Ta) - 4 P[0, 3] + 4 e^(3 Ta) P[0, 3] + 3 P[0, 4] + 3 P[2, 4] + P[4, 4])))}}}}
```

Instant Yule

```
In[ ]:= sample = Table[{1}, {i, 1, 4}];
        substitute[allpees_List] :=
          Total[δ * Peln[#[[1]], #[[2]], Total[#], s, r, Ne, M] & /@ allpees] → δ;
        allFs = Table[partitions[i], {i, 2, Length[sample]}];
        substituteL = substitute /@ allFs;
        (*addition to sum probabilities to right value*)
        (*substituteL =#→sumP[#[[1,2,3]]]*δ&/@substituteL *)

        substituteL
Out[ ]= {δ Peln[0, 0, 2, s, r, Ne, M] + δ Peln[0, 1, 2, s, r, Ne, M] + δ Peln[0, 2, 2, s, r, Ne, M] +
      δ Peln[1, 0, 2, s, r, Ne, M] + δ Peln[1, 1, 2, s, r, Ne, M] + δ Peln[2, 0, 2, s, r, Ne, M] → δ,
  δ Peln[0, 0, 3, s, r, Ne, M] + δ Peln[0, 1, 3, s, r, Ne, M] + δ Peln[0, 2, 3, s, r, Ne, M] +
      δ Peln[0, 3, 3, s, r, Ne, M] + δ Peln[1, 0, 3, s, r, Ne, M] +
      δ Peln[1, 1, 3, s, r, Ne, M] + δ Peln[1, 2, 3, s, r, Ne, M] + δ Peln[2, 0, 3, s, r, Ne, M] +
      δ Peln[2, 1, 3, s, r, Ne, M] + δ Peln[3, 0, 3, s, r, Ne, M] → δ,
  δ Peln[0, 0, 4, s, r, Ne, M] + δ Peln[0, 1, 4, s, r, Ne, M] + δ Peln[0, 2, 4, s, r, Ne, M] +
      δ Peln[0, 3, 4, s, r, Ne, M] + δ Peln[0, 4, 4, s, r, Ne, M] + δ Peln[1, 0, 4, s, r, Ne, M] +
      δ Peln[1, 1, 4, s, r, Ne, M] + δ Peln[1, 2, 4, s, r, Ne, M] + δ Peln[1, 3, 4, s, r, Ne, M] +
      δ Peln[2, 0, 4, s, r, Ne, M] + δ Peln[2, 1, 4, s, r, Ne, M] + δ Peln[2, 2, 4, s, r, Ne, M] +
      δ Peln[3, 0, 4, s, r, Ne, M] + δ Peln[3, 1, 4, s, r, Ne, M] + δ Peln[4, 0, 4, s, r, Ne, M] → δ}

In[ ]:= gfDeltaListYule = Expand[GFaiYule[s, r, Ne, M, ω, sample, {δ, ϵ}]];
        gfDeltaListYule⟦0⟧ = List;
        gfDeltaListYule = gfDeltaListYule //. substituteL;
```

```
In[ ]:= yulePDFS = MakeMargeList[gfDeltaListYule, sample, ω, t, δ, Td, ϵ, Te];
        yulePDFS = Total[#] & /@ yulePDFS;

In[ ]:= yuleCDFS = MakeCumuList[gfDeltaListYule, sample, ω, t, δ, Td, ϵ, Te];
        yuleCDFS = Total[#] & /@ yuleCDFS;
```

Data & Approximation Comparison

```
localPath = SetDirectory[NotebookDirectory[]]
pathToSims = "/simulations/ai_marginals/"
```

Data & Approximation Comparison

```
In[ ]:= recpersite = 1. * 10^-6;
        winDist = 100;
        datNe = 10 000;
        shape = {10 000, 250, 3}; (* {dataPoints, recPoints, infoCols}*)

In[ ]:= sweepTimes = {0, .1, .25, .5, 1.0, 1.5, 2.0} * datNe
Out[ ]= {0, 1000., 2500., 5000., 10 000., 15 000., 20 000.}

In[ ]:= localPath = SetDirectory[NotebookDirectory[]];
        pathToSims = localPath <> "/simulations/ai_marginals/";
        dataFiles = {"np_s4_0.dat", "np_s4_5000.dat", "np_s4_10000.dat",
           "np_s4_15000.dat", "np_s4_17500.dat", "np_s4_19000.dat", "np_s4_20000.dat"};
        dataFiles = Reverse[dataFiles]; (*do this because the times listed in the fiels
          are ms times not sample times, their reverse order matches the sweep times*)
        dataFiles = pathToSims <> # & /@ dataFiles

In[ ]:= allData = MapThread[MakeDataArray[#1, #2, shape, datNe] &, {sweepTimes, dataFiles}];
```

```
In[ ]:= plt0[idxT_, idxr_, ss_, nn_, Tai_, Tdonor_] := Module[
          {rec = recpersite * idxr * winDist, m = Floor[2 nn 2 ss], Ta = Tai, Ts = Tdonor, α},
          α = rec / ss Log[2 nn ss];
          Show[
           Plot[
            Evaluate[yulePDFS /. {s → ss, r → rec, Ne → nn, M → m, Td → Ta, Te → Ts} /. Peln → PELN],
            {t, 0, 3}, PlotStyle → Evaluate[Darker[#] & /@ {Blue, Green, Red}],
            Exclusions → None, PlotRange → Full],
           Plot[Evaluate[#[t, α, Ta, Ts] & /@ funPdfs], {t, 0, 3}, PlotRange → Full, PlotStyle →
             Evaluate[{Darker[#], Dashed} & /@ {Blue, Green, Red}], Exclusions → None],
           Histogram[allData[[idxT]][[idxr]][[2]], {.05}, "PDF",
            ChartStyle → Evaluate[Lighter[#] & /@ {Blue, Green, Red}]
           ],
           PlotLabel → "α= " <> ToString[α] <> "   Ta= " <> ToString[Ta],
           PlotRange → {{0, 3}, {0, 2}}
          ]
         ]
```

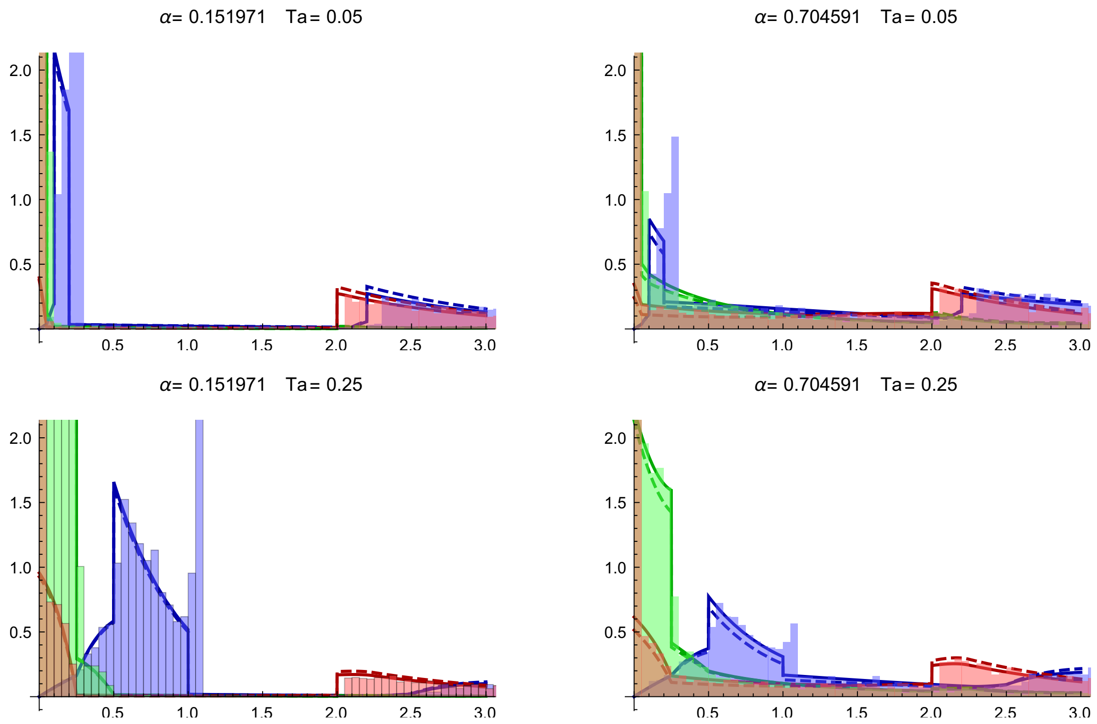
```
In[ ]:= plt1[idxT_, idxr_, ss_, nn_, Tai_, Tdonor_] := Module[
          {rec = recpersite * idxr * winDist, m = Floor[2 nn 2 ss], Ta = Tai, Ts = Tdonor, α},
          α = rec / ss Log[2 nn ss];
          Show[
           Plot[
            Evaluate[yuleCDFS /. {s → ss, r → rec, Ne → nn, M → m, Td → Ta, Te → Ts} /. Peln → PELN],
            {t, 0, 3}, PlotStyle → Evaluate[Darker[#] & /@ {Blue, Green, Red}],
            Exclusions → None, PlotRange → Full],
           Plot[Evaluate[#[t, α, Ta, Ts] & /@ funCdfs], {t, 0, 3}, PlotRange → Full, PlotStyle →
             Evaluate[{Darker[#], Dashed} & /@ {Blue, Green, Red}], Exclusions → None],
           Histogram[allData[[idxT]][[idxr]][[2]], {.05}, "CDF",
            ChartStyle → Evaluate[Lighter[#] & /@ {Blue, Green, Red}]
           ],
           PlotLabel → "α= " <> ToString[α] <> "   Ta= " <> ToString[Ta],
           PlotRange → {{0, 3}, {0, 1}}
          ]
         ]
```

```
In[ ]:= (*sampleTimeIndexes = {2, 4, 5, 7};
        distanceIndexes = {3, 11, 51, 101};*)
        sampleTimeIndexes = {2, 4};
        distanceIndexes = { 11, 51};
```

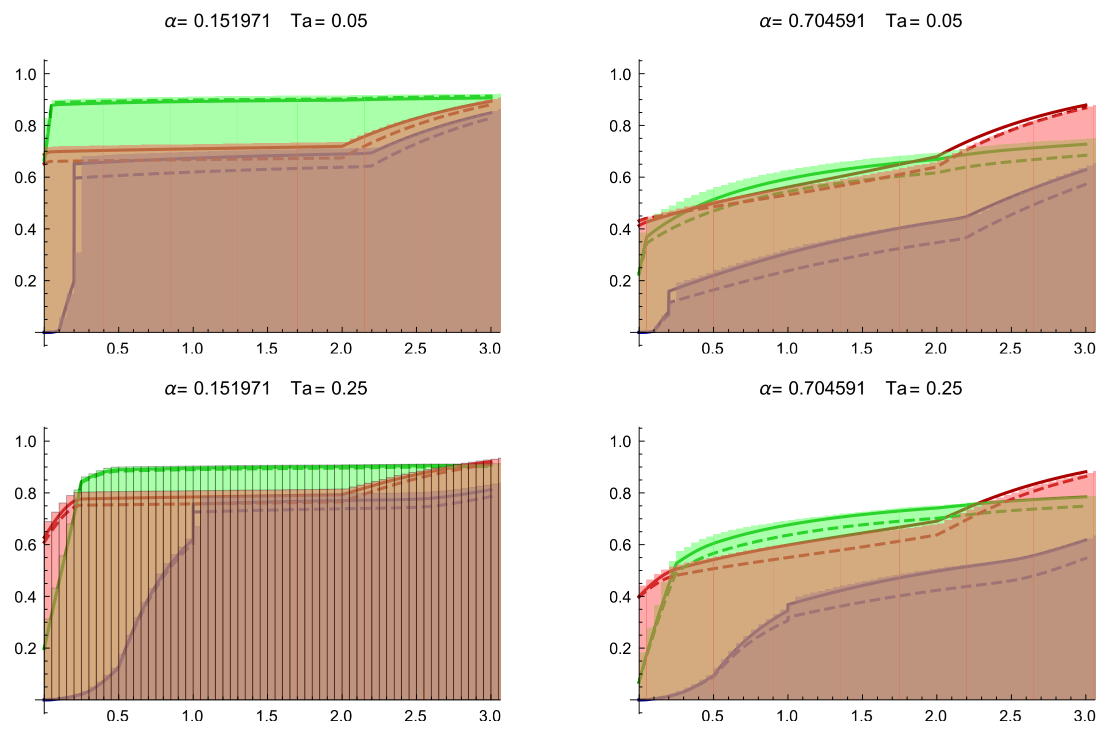
```
In[ ]:= sampleTimeIndexes = {1, 3, 5, 7};
        distanceIndexes = {2, 10, 50, 100};
```

For the classic hard sweep scenario, there is very little improvement in the accuracy of the analytic results using the Yule approximation despite a substantial increase in computation costs. Despite only minor differences, we observed more accurate parameter estimation using the Yule approximation for the classic sweep scenario. For the adaptive introgression scenario, however, the Yule approximation does provide a clearly better fit for the marginal distributions, particularly at intermediate recombina - tion distances where lineages have the highest probability of being separated during the time prior to the sweep and before the split of the ancestral population (this is easiest to see in the CDFs). The Yule approximation is likely more useful when estimating parameters under models with more complex demography.

Out[∘]=



Out[ ]=



Site Frequency Spectrum

Here we derive the SFS, analgous to the calssic hard sweep scenario of the main text. The GF as a list and the mean length of iton branches have been pre-computed.

```
(*n=8;
gfStarList = Block[{sample = Table[{1},{i,1,n}],gfDeltaEpsilon },
  gfDeltaEpsilon =GFaiStar[α,ω,sample ,δ,ϵ] //Expand;
  gfDeltaEpsilon [[0]] = List;
  gfDeltaEpsilon = SubSimplifyPeeRulesv2 [#,n]&/@gfDeltaEpsilon //Parallelize ;
  gfDeltaEpsilon ];

meansList = MakeMeanList [gfStarList ,n,ω,t,δ,Ta,ϵ,Ts];
meanItonFunList =
 Function [{α,Ta,Ts},Evaluate [#//.P→PknToAlpha /.{Td→Ta,Te→Ts}]]&/@meansList ;
sfs[α_,Ta_,Ts_]:=Block[{a = #[α,Ta,Ts]&/@meanItonFunList },a/Total[a]]*)
```

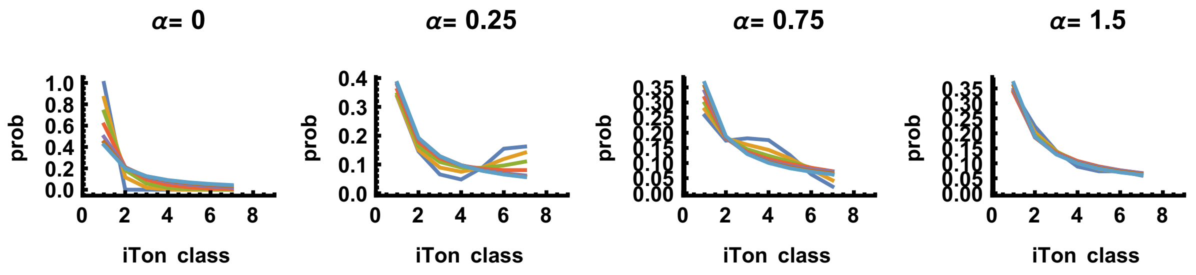
```
In[ ]:= SetDirectory[NotebookDirectory[]];
        (*DumpSave["ai_iTonBL_8_gfStarList_meansList.mx",{gfStarList,meansList}]*)
        Get["ai_iTonBL_8_gfStarList_meansList.mx"]
        meanItonFunList =
          Function[{α, Ta, Ts}, Evaluate[# //. P → PknToAlpha /. {Td → Ta, Te → Ts}]] & /@ meansList;
        sfs[α_, Ta_, Ts_] := Block[{a = #[α, Ta, Ts] & /@ meanItonFunList}, a / Total[a]]

In[ ]:= sfs[.1, .2, 3]

Out[ ]= {0.396556, 0.150744, 0.0763606, 0.0549814, 0.0639682, 0.0983329, 0.159057}

Out[ ]= {α= 0, α= 0.25, α= 0.75, α= 1.5}

Out[ ]=
```


α= 0
α= 0.25
α= 0.75
α= 1.5
prob
iTon class