```
SetDirectory @NotebookDirectory [];
```

Generating Function Functions

Functions to obtain the generating function via recursion. We use addition to represent coalesent states of ancestral lineages. This works because addition is associative and allows simplification of the branch indices as much as possible along the way. That means we either pass a fully-labelled set {{a},{b},{c}} and obtain the full generating function for branches of type {a+b} and {c}. If we pass {{1},{1},{1}} we obtain i-Ton labelled branch types, e.g., {2}, {1}.

The machinery for the recursion (below) will provide the generating function for a fully labelled sub-set, which we examine for a sample of n=2 lineages. However, the remaining functions are used to simplify and automate analysis of the i-Ton labelled branches and are defined for a sample of n>2 lineages.

Neutral Coalescent

Starlike Approximation

Instant Yule Approximation

Functions for Data

Results

n=2

full label

In[ ]:=
```
sample = {{a}, {b}}
```

Out[ ]= {{a}, {b}}

In[ ]:=
```
neuGF = ConvertNotation[GFn[ω, sample], ω]
(*note that convert notation is not actually needed in the n=2 case*)
```

Out[ ]=
$$\frac{1}{1+\omega[\{a\}]+\omega[\{b\}]}$$

In[ ]:=
```
gfSelDelta = GFstar[α, ω, sample, {δ}] // Expand
 (*expanded out, each term of sum is a unique topology*)
```

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

In[ ]:=
```
gfSelDelta = gfSelDelta //. subAllPeeToEpsilon[2];
gfSelDeltaList = gfSelDelta;
gfSelDeltaList[[0]] = List;
gfSelDeltaList
(*Simplification by summing P[i,k] terms. obtain the gf list-wise by topology.*)
```

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

In[ ]:=
```
gfSelTaList = InverseLaplaceTransform[#/δ, δ, Ta] &/@ gfSelDeltaList;
(*take inverse laplace transform separately for each toplogy*)
gfSelTaList
```

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

In[ ]:=
```
(*path probabilities*)
gfSelTaList /. ω[_] -> 0 // Simplify
```

Out[ ]=
$$\{1-e^{-Ta},\ e^{-Ta}\,P[0,2],\ e^{-Ta}\,P[1,2],\ e^{-Ta}\,P[2,2]\}$$

time to the most recent common ancestor.

Substitution to get i-ton labeled genelogy, here resulting in the singleton branch lengths. We also halve ω to get Tmrca, rather than the singleton branch length, in order to have a dual measure for genetic diversity.

In[ ]:=
```
tmrcaNeuGF = neuGF //. {ω[{a}] → ω / 2, ω[{b}] → ω / 2}
tmrcaSelGFDelta = GFstar[α, ω, sample, {δ}] /. {ω[{a}] → ω / 2, ω[{b}] → ω / 2}
tmrcaSelGFTa = InverseLaplaceTransform [tmrcaSelGFDelta / δ, δ, Ta] //. subAllPeeToOne [2]
```

Out[ ]=
$$\frac{1}{1+\omega}$$

Out[ ]=
$$\frac{1+\delta\,P[0,2]+\frac{\delta\,P[1,2]}{1+\omega}+\frac{\delta\,P[2,2]}{1+\omega}}{1+\omega+\delta\,P[0,2]+\delta\,P[1,2]+\delta\,P[2,2]}$$

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

expected time to most recent common ancestor

In[ ]:=
```
neuMeanTmrca = Limit[(-1) D[tmrcaNeuGF , ω], {ω → 0}]
```

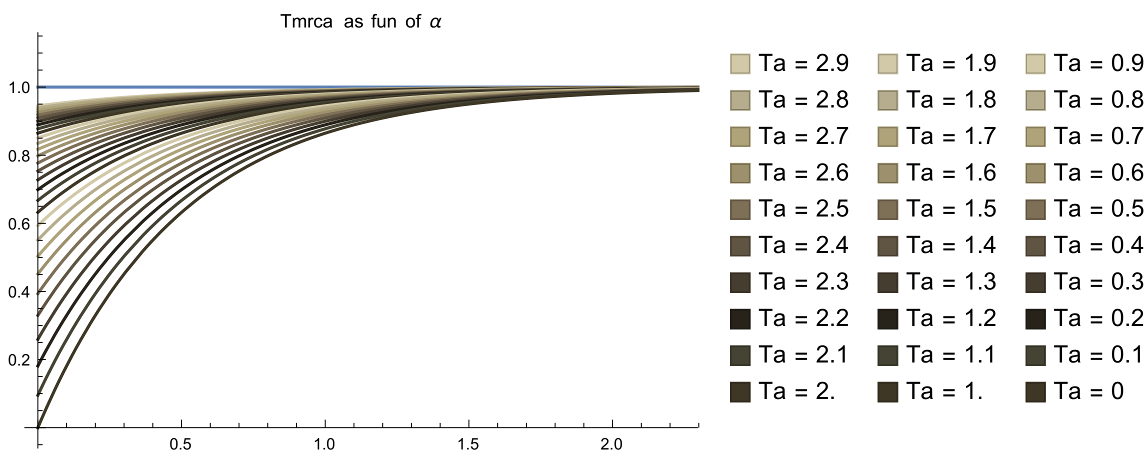
Out[ ]= 1

In[ ]:=
```
selMeanTmrca = Limit[(-1) D[tmrcaSelGFTa , ω], {ω → 0}]
selMeanTmrca = selMeanTmrca /. P → PknToAlpha
```

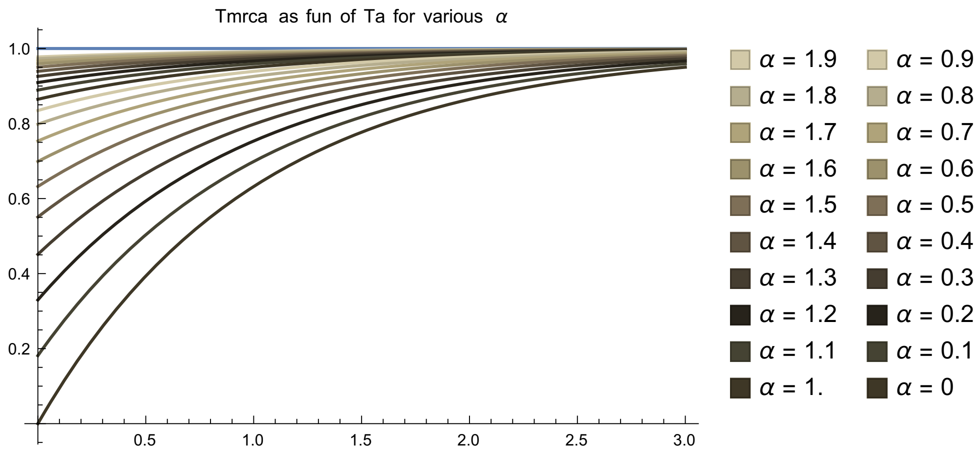
Out[ ]= $1-e^{-Ta}\,P[0,2]$

Out[ ]= $1-e^{-Ta-2\,\alpha}$

Out[ ]=



Out[∘]=



distribution of time to most recent common ancestor

In[∘]:=
```
neuPDFTmrca = InverseLaplaceTransform [tmrcaNeuGF , ω, t]
```

Out[∘]= e^{-t}

In[∘]:=
```
neuCDFTmrca = InverseLaplaceTransform [tmrcaNeuGF / ω, ω, t]
```

Out[∘]= $1 - e^{-t}$

In[∘]:=
```
selPDFTmrca = InverseLaplaceTransform [tmrcaSelGFTa , ω, t]
```

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

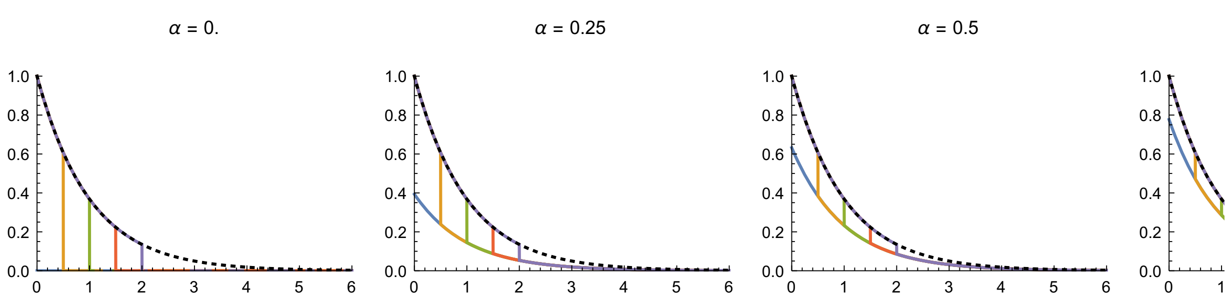
In[∘]:=
```
selCDFTmrca = InverseLaplaceTransform [tmrcaSelGFTa / ω, ω, t]
```

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

In[∘]:=
```
selPDFTmrca = selPDFTmrca /. P → PknToAlpha ;
(*substituting to get as function of α for plotting*)
selCDFTmrca = selCDFTmrca /. P → PknToAlpha ;
```

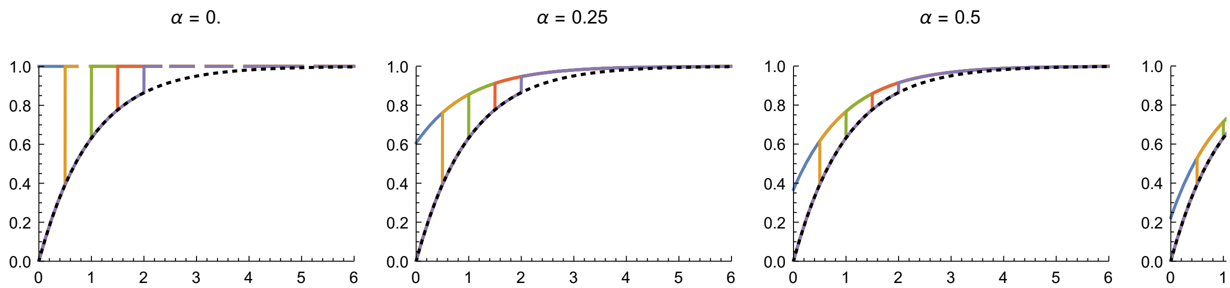
PDF

Out[∘]=



CDF

Out[ ]= $e^{-t}\left(-1+e^{t}+e^{-2\alpha}\,\text{HeavisideTheta}[t-\text{Ta}]\right)$

Out[ ]=


Determining size of the pointmass at t = Ta:

```
In[ ]:= lower = Limit[selCDFTmrca, {t → Ta}, Direction → "FromBelow"]
upper = Limit[selCDFTmrca, {t → Ta}, Direction -> "FromAbove"]
pointMassSize = upper - lower // FullSimplify
```

Out[ ]= $1-e^{-\text{Ta}}$

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

Out[ ]= $e^{-\text{Ta}}\,\text{P}[0,2]$

n = 3 iton labeled section

Note that the MakeMeanList-type functions now require two dummy variables, which will be needed for the adaptive introgression section

```
In[ ]:= n = 3;
```

neutral and starlike GF

```
In[ ]:= neugfList = GFn[ω, Table[{1}, {i, 1, n}]] // Expand; neugfList[[0]] = List;
```

```
In[ ]:= nmList = MakeMeanList[neugfList, n, ω, t, δ, Ta];
neuMeans = Function[{α, Ta}, Evaluate[# /. P → PknToAlpha]] & /@ nmList;
npList = MakePDFList[neugfList, n, ω, t, δ, Ta];
neuPdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ npList;
ncList = MakeCDFList[neugfList, n, ω, t, δ, Ta];
neuCdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ ncList;
```

In[ ]:=
```
nmList
npList // FullSimplify
ncList // FullSimplify
```

Out[ ]= {2, 1}

Out[ ]= $\left\{6\,e^{-t} - \frac{3}{4}\,e^{-3\,t/2}\,(8 + 3\,t),\ \frac{1}{5}\,e^{-3\,t}\left(9 + e^{5\,t/2}\right)\right\}$

Out[ ]= $\left\{1 - 6\,e^{-t} + e^{-3\,t/2}\left(5 + \frac{3\,t}{2}\right),\ 1 - \frac{1}{5}\,e^{-3\,t}\left(3 + 2\,e^{5\,t/2}\right)\right\}$

In[ ]:=
```
selgfDeltaList = GetGfAsList[n];
selgfTsList = InverseLaplaceTransform [1 / δ #, δ, Ts] & /@ selgfDeltaList ;
```

In[ ]:=
```
smList = MakeMeanList[selgfDeltaList , n, ω, t, δ, Ta];
selMeans = Function[{α, Ta}, Evaluate[# /. P → PknToAlpha]] & /@ smList;
spList = MakePDFList[selgfDeltaList , n, ω, t, δ, Ta];
selPdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ spList;
scList = MakeCDFList[selgfDeltaList , n, ω, t, δ, Ta];
selCdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ scList;
```

PDFs for neutral case piecewise, first is singletons, second is doubletons, third is tripletons.

In[ ]:=
```
npListPieceWise = {MakePiecewise [#, t, Ta == 0], MakePiecewise [#, t, Ta > 0]} & /@ npList;
Print@Piecewise[Flatten[#, 1]] & /@ npListPieceWise
```

$$\begin{cases} 6\,e^{-t} - \frac{3}{4}\,e^{-3\,t/2}\,(8 + 3\,t) & 0 \le t < \infty \\ 0 & \text{True} \end{cases}$$

$$\begin{cases} \frac{1}{5}\,e^{-3\,t}\left(9 + e^{5\,t/2}\right) & 0 \le t < \infty \\ 0 & \text{True} \end{cases}$$

Out[ ]= {Null, Null}

PDFs for selective case, ordered by 1-Ton, 2-Ton, 3-Ton. In each, separate the case where Ta=0 from Ta>0.

In[ ]:=
```
spListPieceWise = {MakePiecewise [#, t, Ta == 0], MakePiecewise [#, t, Ta > 0]} & /@ spList;
Print@Piecewise[Flatten[#, 1]] & /@ spListPieceWise // FullSimplify
```

$$\begin{cases} \text{DiracDelta}[t]\,P[0,3] + e^{-t}(P[1,3] + t\,(P[2,3] + P[3,3])) \\ e^{-t}\,t \\ \frac{1}{2}e^{-t}(2t + 3(1 - t + Ta)\,P[0,2]) \\ e^{-t}\,\text{DiracDelta}[t - 3\,Ta]\,P[0,3] + e^{-t}(P[1,3] + 3\,Ta\,(P[1,2] + P[2,2] - P[2,3] - P[3,3]) + t\,(P[2,3] + P[3,3])) \\ 0 \end{cases}$$

$$\begin{cases} e^{-t}(1 - P[0,3]) + \text{DiracDelta}[t]\,P[0,3] & Ta == 0 \,\&\&\, 0 \le t < \infty \\ e^{-t-3\,Ta}(e^{3\,Ta} + P[0,2] + 2e^{3t}P[0,2] - P[0,3]) + e^{-3\,Ta}\,\text{DiracDelta}[t]\,P[0,3] & Ta > 0 \,\&\&\, 0 \le t < Ta \\ e^{-t-3\,Ta}(-e^{3\,Ta}(-1 + P[0,2]) + P[0,2] - P[0,3]) & Ta > 0 \,\&\&\, Ta \le t < \infty \\ 0 & \text{True} \end{cases}$$

Out[ ]= {Null, Null}

In[ ]:= **GetJumps[npList, ncList, t][[3]] (*note that when there are no discontinuitites, it prints nonsense back. here, only the 3-Ton branches have a discontinuitiy *)**

Out[ ]= $\left\{\left\{1, \{\}, \left\{\left\{\{\}, \text{Limit}\left[1 - 6e^{-t} + e^{-3t/2}\left(5 + \frac{3t}{2}\right), \text{MakeALim}[\{\}, t], \text{Direction} \to \text{FromAbove}\right] - \text{Limit}\left[1 - 6e^{-t} + e^{-3t/2}\left(5 + \frac{3t}{2}\right), \text{MakeALim}[\{\}, t], \text{Direction} \to \text{FromBelow}\right]\right\}\right\}\right\}, \left\{2, \{\}, \left\{\left\{\{\}, \text{Limit}\left[1 - \frac{1}{5}e^{-3t}(3 + 2e^{5t/2}), \text{MakeALim}[\{\}, t], \text{Direction} \to \text{FromAbove}\right] - \text{Limit}\left[1 - \frac{1}{5}e^{-3t}(3 + 2e^{5t/2}), \text{MakeALim}[\{\}, t], \text{Direction} \to \text{FromBelow}\right]\right\}\right\}\right\}\right\}[[3]]$

In[ ]:= **Print[#] &/@ GetJumps[spList, scList, t]**

{1, {HeavisideTheta [t - 3 Ta], HeavisideTheta [t - Ta], DiracDelta [t - 3 Ta]}, {{t - 3 Ta, $e^{-3\,Ta}$ P[0, 3]}}}

{2, {HeavisideTheta [t - Ta], DiracDelta [t]}, {{t, $e^{-3\,Ta}$ P[0, 3]}}}

Out[ ]= {Null, Null}

n = 4 iton labeled section

Note that the MakeMeanList-type functions now require two dummy variables, which will be needed for the adaptive introgression section

In[ ]:= **n = 4;**

neutral and starlike GF

In[ ]:= **neugfList = GFn[ω, Table[{1}, {i, 1, n}]] // Expand; neugfList[[0]] = List;**

In[ ]:=
```
nmList = MakeMeanList[neugfList, n, ω, t, δ, Ta];
neuMeans = Function[{α, Ta}, Evaluate[# /. P → PknToAlpha]] & /@ nmList;
npList = MakePDFList[neugfList, n, ω, t, δ, Ta];
neuPdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ npList;
ncList = MakeCDFList[neugfList, n, ω, t, δ, Ta];
neuCdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ ncList;
```

In[ ]:=
```
nmList
npList // FullSimplify
ncList // FullSimplify
```

Out[ ]= $\left\{2, 1, \frac{2}{3}\right\}$

Out[ ]= $\left\{6\, e^{-t} - \frac{3}{4} e^{-3\,t/2} (8 + 3\,t),\ \frac{1}{5} e^{-3\,t} \left(9 + e^{5\,t/2}\right),\ \frac{2\, e^{-t}}{3} + \frac{\text{DiracDelta}[t]}{3}\right\}$

Out[ ]= $\left\{1 - 6\, e^{-t} + e^{-3\,t/2} \left(5 + \frac{3\,t}{2}\right),\ 1 - \frac{1}{5} e^{-3\,t} \left(3 + 2\, e^{5\,t/2}\right),\ 1 - \frac{2\, e^{-t}}{3}\right\}$

In[ ]:=
```
selgfDeltaList = GetGfAsList[n];
selgfTsList = InverseLaplaceTransform[1/δ #, δ, Ts] & /@ selgfDeltaList;
```

In[ ]:=
```
smList = MakeMeanList[selgfDeltaList, n, ω, t, δ, Ta];
selMeans = Function[{α, Ta}, Evaluate[# /. P → PknToAlpha]] & /@ smList;
spList = MakePDFList[selgfDeltaList, n, ω, t, δ, Ta];
selPdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ spList;
scList = MakeCDFList[selgfDeltaList, n, ω, t, δ, Ta];
selCdfs = Function[{α, Ta, t}, Evaluate[# /. P → PknToAlpha]] & /@ scList;
```

PDFs for neutral case piecewise, first is singletons, second is doubletons, third is tripletons.

In[ ]:=
```
npListPieceWise = {MakePiecewise[#, t, Ta == 0], MakePiecewise[#, t, Ta > 0]} & /@ npList;
Print@Piecewise[Flatten[#, 1]] & /@ npListPieceWise
```

$$\begin{cases} 6\, e^{-t} - \frac{3}{4} e^{-3\,t/2} (8 + 3\,t) & 0 \le t < \infty \\ 0 & \text{True} \end{cases}$$

$$\begin{cases} \frac{1}{5} e^{-3\,t} \left(9 + e^{5\,t/2}\right) & 0 \le t < \infty \\ 0 & \text{True} \end{cases}$$

$$\begin{cases} \frac{2\, e^{-t}}{3} + \frac{\text{DiracDelta}[t]}{3} & (\text{Ta} == 0\ \&\&\ 0 \le t < \infty) \,||\, (\text{Ta} > 0\ \&\&\ 0 \le t < \infty) \\ 0 & \text{True} \end{cases}$$

Out[ ]= {Null, Null, Null}

PDFs for selective case, ordered by 1-Ton, 2-Ton, 3-Ton. In each, separate the case where Ta=0 from Ta>0.

```
In[ ]:= spListPieceWise = {MakePiecewise[#, t, Ta == 0], MakePiecewise[#, t, Ta > 0]} & /@ spList;
        Print@Piecewise[Flatten[#, 1]] & /@ spListPieceWise
```

$$
\begin{cases}
\text{DiracDelta}[t]\,P[0,4]+\frac{1}{4}e^{-3t/2}\Big(-6P[2,4]-3(8+3t)(P[3,4]+P[4,4])+4e^{t/2}(P[1,4]+2P[2,4]+6(P[3,4]+P[4,4]))\Big) & Ta==0\,\&\&\,0\le t<\infty\\
\frac{3}{4}e^{-3t/2}\left(-8+8e^{t/2}-3t+4e^{\frac{t}{4}-Ta}\left(-1+e^{t/4}\right)(-1+P[0,2]+P[1,2]+P[2,2])\right) & Ta>0\,\&\&\,0\le t<Ta\\
\frac{3}{4}e^{-2t-Ta}\Big(-16e^{2Ta}(1+P[0,2]-P[1,2]-P[2,2])+4e^{3t/4}\left(-1+e^{t/4}\right)(-1+P[0,2]+P[1,2]+P[2,2])+8e^{\frac{1}{3}(t+5Ta)}(2P[0,2]-3(-1+P[1,2]+P[2,2]))+e^{\frac{t}{2}+Ta}\left(-8-3t+8e^{t/2}(P[1,2]+P[2,2])\right)\Big) & Ta>0\,\&\&\,Ta\le t<2Ta\\
6e^{-t}(P[1,2]+P[2,2])-3e^{-\frac{5t}{4}-Ta}(-1+P[0,2]+P[1,2]+P[2,2])+6e^{-\frac{5t}{3}+\frac{2Ta}{3}}(2P[0,2]-3(-1+P[1,2]+P[2,2]))+4e^{-t-Ta}(-3P[1,2]+P[1,3]+3(-P[2,2]+P[2,3]+P[3,3]))+\frac{3}{4}e^{-3t/2}\big(t(3-6P[2,3]-6P[3,3])+4(-7-3P[0,2]+P[0,3]+9P[1,2]-P[1,3]+9P[2,2]-4P[2,3]-4P[3,3]+3Ta(-1+P[2,3]+P[3,3]))\big) & Ta>0\,\&\&\,2Ta\le t<4Ta\\
e^{-3t/2}\,\text{DiracDelta}[t-4Ta]\,P[0,4]+6e^{-t}(P[1,2]+P[2,2])+4e^{-t-Ta}(-3P[1,2]+P[1,3]+3(-P[2,2]+P[2,3]+P[3,3]))-\frac{3}{4}e^{-3t/2}\big(2P[2,4]+12Ta(P[2,3]+P[3,3]-P[3,4]-P[4,4])+(8+3t)(P[3,4]+P[4,4])\big)+e^{-t-2Ta}\big(6P[1,2]-4P[1,3]+P[1,4]+2(3P[2,2]-6P[2,3]+P[2,4]+3(-2P[3,3]+P[3,4]+P[4,4]))\big) & Ta>0\,\&\&\,4Ta\le t<\infty\\
0 & \text{True}
\end{cases}
$$

$$
\begin{cases}
\text{DiracDelta}[t]\,P[0,4]+\text{DiracDelta}[t]\,P[1,4]+\frac{1}{5}e^{-3t}\left(9+e^{5t/2}\right)(P[2,4]+P[3,4]+P[4,4]) & Ta==0\,\&\&\,0\le t<\infty\\
e^{-6Ta}\,\text{DiracDelta}[t]\,P[0,4]+e^{-6Ta}\,\text{DiracDelta}[t]\,P[1,4]+\frac{1}{35}e^{-3(t+2Ta)}\Big(-175e^{11t/2}P[0,2]+15e^{6t}(12P[0,2]+14P[0,3]+5(-1+2P[1,3]+P[2,3]+P[3,3]))+63\left(e^{6Ta}-P[2,3]+P[2,4]-P[3,3]+P[3,4]+P[4,4]\right)+e^{5t/2}\left(5+7e^{6Ta}-5P[0,2]-10P[1,3]-12P[2,3]+7P[2,4]-12P[3,3]+7P[3,4]+7P[4,4]\right)\Big) & Ta>0\,\&\&\,0\le t<Ta\\
\frac{1}{35}e^{-3(t+2Ta)}\Big(168e^{5t+Ta}P[0,2]+2e^{\frac{5t}{2}+\frac{7Ta}{2}}(6P[0,2]+5P[1,3]+6(-1+P[2,3]+P[3,3]))+7e^{6Ta}\left(e^{5t/2}+9(P[2,3]+P[3,3])\right)-63(P[2,3]-P[2,4]+P[3,3]-P[3,4]-P[4,4])+e^{5t/2}\left(5-5\left(1+35e^{3t}\right)P[0,2]-10P[1,3]-12P[2,3]+7P[2,4]-12P[3,3]+7P[3,4]+7P[4,4]\right)\Big) & Ta>0\,\&\&\,Ta\le t<2Ta\\
\frac{1}{35}e^{-3(t+2Ta)}\Big(2e^{\frac{5t}{2}+\frac{7Ta}{2}}(6P[0,2]+5P[1,3]+6(-1+P[2,3]+P[3,3]))+7e^{6Ta}\left(-e^{5t/2}(-1+P[0,2])+9(P[2,3]+P[3,3])\right)-63(P[2,3]-P[2,4]+P[3,3]-P[3,4]-P[4,4])+e^{5t/2}\left(5-5P[0,2]-10P[1,3]-12P[2,3]+7P[2,4]-12P[3,3]+7P[3,4]+7P[4,4]\right)\Big) & Ta>0\,\&\&\,2Ta\le t<\infty\\
0 & \text{True}
\end{cases}
$$

$$\begin{cases} \text{DiracDelta}[t]\,P[0,4] + \frac{1}{3}\text{DiracDelta}[t]\,P[2,4] + \frac{1}{3}\text{DiracDelta}[t]\,P[3,4] + \\ \quad \frac{1}{3}\text{DiracDelta}[t]\,P[4,4] + \frac{1}{3}e^{-t}(3\,P[1,4] + 2\,(P[2,4] + P[3,4] + P[4,4])) & \text{Ta} == 0 \,\&\&\, 0 \le t < \infty \\ \frac{\text{DiracDelta}[t]}{3} + \frac{1}{3}e^{-6\,\text{Ta}}\,\text{DiracDelta}[t] - \frac{2}{3}e^{-3\,\text{Ta}}\,\text{DiracDelta}[t] + \\ \quad \frac{2}{3}e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,(-1 + P[0,3]) - \frac{2}{3}e^{-3\,\text{Ta}}\,\text{DiracDelta}[t]\,(-1 + P[0,3]) - \\ \quad 2\,e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,P[0,3] + 2\,e^{-3\,\text{Ta}}\,\text{DiracDelta}[t]\,P[0,3] + \\ \quad e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,P[0,4] + \frac{1}{3}e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,P[2,4] + \\ \quad \frac{1}{3}e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,P[3,4] + \frac{1}{3}e^{-6\,\text{Ta}}\,\text{DiracDelta}[t]\,P[4,4] + \\ \quad \frac{1}{3}e^{-t-6\,\text{Ta}}\,(-2\,P[0,2] + 2\,(e^{6\,\text{Ta}} - 5\,e^{6\,t}\,P[0,2] + 2\,e^{3\,\text{Ta}}\,((1 + 2\,e^{3\,t})\,P[0,2] - P[0,3])) + \\ \qquad 4\,P[0,3] + 3\,P[1,4] + 2\,(-1 + P[2,4] + P[3,4] + P[4,4])) & \text{Ta} > 0 \,\&\&\, 0 \le t < \text{Ta} \\ \frac{1}{3}e^{-t-6\,\text{Ta}}\,(4\,e^{3\,\text{Ta}}\,(P[0,2] - P[0,3]) + 4\,P[0,3] + 2\,P[1,2] + 3\,P[1,4] + \\ \qquad 2\,e^{6\,\text{Ta}}\,(P[1,2] + P[2,2]) + 2\,(-2 + P[2,2] + P[2,4] + P[3,4] + P[4,4])) & \text{Ta} > 0 \,\&\&\, \text{Ta} \le t < \infty \\ 0 & \text{True} \end{cases}$$

Out[ ]= {Null, Null, Null}

In[ ]:= **GetJumps[npList, ncList, t][[3]] (*note that when there are no discontinuitites, it prints nonsense back. here, only the 3-Ton branches have a discontinuitiy *)**

In[ ]:= **Print[#] & /@ GetJumps[spList, scList, t]**

$\{1, \{\text{HeavisideTheta}[t - 4\,\text{Ta}], \text{HeavisideTheta}[t - 2\,\text{Ta}], \text{HeavisideTheta}[t - \text{Ta}], \text{DiracDelta}[t - 4\,\text{Ta}]\}, \{\{t - 4\,\text{Ta}, e^{-6\,\text{Ta}}\,P[0,4]\}\}\}$

$\{2, \{\text{HeavisideTheta}[t - \text{Ta}], \text{HeavisideTheta}[t - 2\,\text{Ta}], \text{DiracDelta}[t]\}, \{\{t, e^{-6\,\text{Ta}}\,(P[0,4] + P[1,4])\}\}\}$

$$\left\{3, \{\text{DiracDelta}[t], \text{HeavisideTheta}[t - \text{Ta}]\}, \left\{\left\{t, \frac{1}{3}e^{-6\,\text{Ta}}\,(e^{6\,\text{Ta}} - 4\,P[0,3] + 4\,e^{3\,\text{Ta}}\,P[0,3] + 2\,P[0,4] - P[1,4])\right\}\right\}\right\}$$

Out[ ]= {Null, Null, Null}

Instant Yule GF

```
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
```

```
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[ ]:= yulegfDeltaList = Expand @GFYule[s, r, Ne, M, ω, sample, {δ}];
        yulegfDeltaList [[0]] = List;
        yulegfDeltaList = yulegfDeltaList //. substituteL ;
```

```
In[ ]:= yulePDFS = MakeMargeList [yulegfDeltaList , sample, ω, δ];
        yulePDFS = Total[#] & /@ yulePDFS ;
```

```
In[ ]:= yuleCDFS = MakeCumuList [yulegfDeltaList , sample, ω, δ];
        yuleCDFS = Total[#] & /@ yuleCDFS ;
```

```
In[ ]:= # /. {s → .05, Ne → 10 000, r → .005, Ta → .1, M → Floor[2 * 10 000 * .05]} /. Peln → PELN & /@
          yuleCDFS(*substitution to get numerical
           values. then can evaluate by changing Peln to PELN*)
```

i-Ton marginals: starlike vs yule approximation

```
recpersite = 1. * 10 ^ -7;
winDist = 1000;
datNe = 10 000;
sweepTimes = {0, .1, .25, .5, 1.0, 1.5, 2.0} * datNe * 2
classicDataShape = {10 000, 250, 3}
```

```
localPath = SetDirectory[NotebookDirectory[]];
pathToSims = localPath <> "/simulations/classic_marginals/";
classicSweepDataFiles = {"np_s4_0.dat",
   "np_s4_0_1.dat",
   "np_s4_0_25.dat",
   "np_s4_0_5.dat",
   "np_s4_1.dat",
   "np_s4_1_5.dat",
   "np_s4_2.dat"
  };
classicSweepDataFiles = pathToSims <> # & /@ classicSweepDataFiles
```

```
classicSweepData = MapThread[MakeDataArray[#1, #2, classicDataShape, datNe] &,
   {sweepTimes, classicSweepDatFiles}];
```

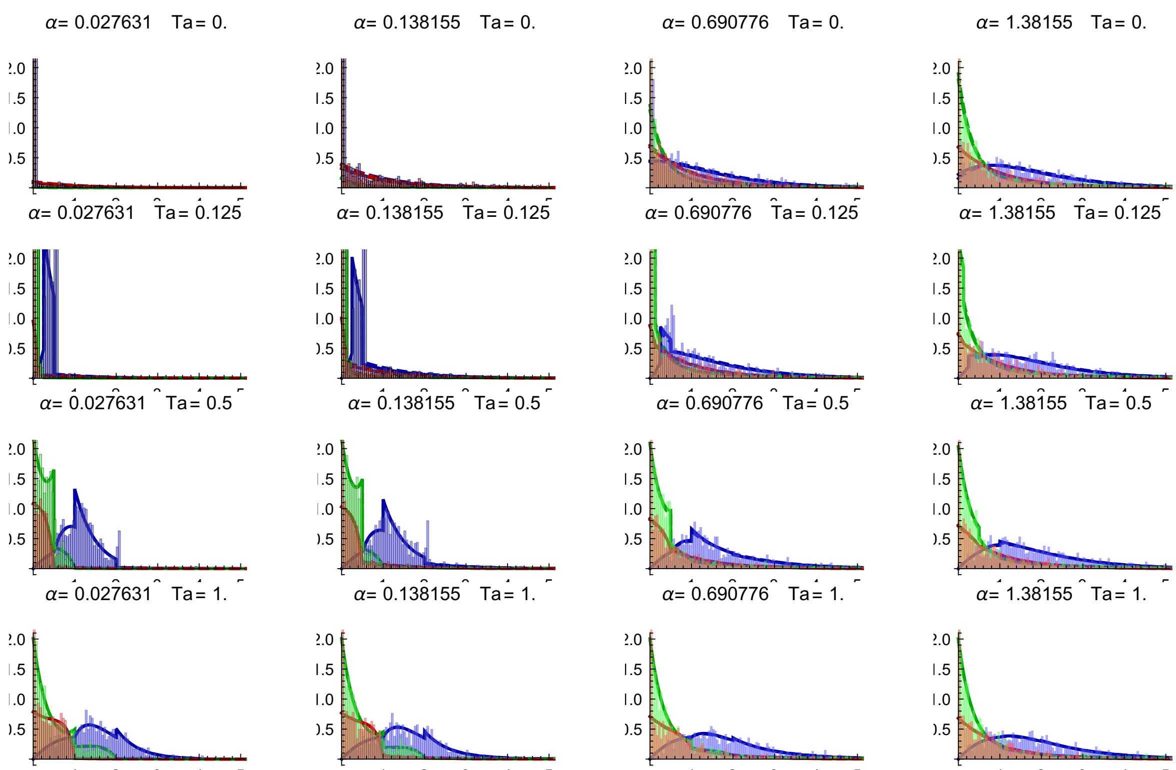
The starlike approximation suffices for classic hard sweeps. We see little difference between the predictions of the starlike (dashed) and yule (solid) predictions in the figures below, if only a slight improvement of the fit of the CDFs using the more-complicated Yule approximations. In contrast, the computation cost increases dramatically when using the Yule approximation.

```
In[ ]:= plt0[idxT_, idxr_, ss_, nn_] := Block[
  {rec = recpersite * idxr * winDist,
   M = Floor[2 nn 2 ss], Td = sweepTimes[[idxT]] / (2 * datNe), α},
  α = rec / ss Log[2 nn ss];
  Show[
   Plot[Evaluate[yulePDFS /. {s → ss, r → rec, Ne → nn, M → M, Ta → Td} /. Peln → PELN],
    {t, 0, 6}, PlotStyle → Evaluate[Darker[#] & /@ {Blue, Green, Red}],
    Exclusions → None, PlotRange → Full],
   Plot[Evaluate[#[α, Td, t] & /@ selPdfs], {t, 0, 6}, PlotRange → Full, PlotStyle →
     Evaluate[{Darker[#], Dashed} & /@ {Blue, Green, Red}], Exclusions → None],
   Histogram[classicSweepData[[idxT]][[idxr]][[2]], {.05}, "PDF",
    ChartStyle → Evaluate[Lighter[#] & /@ {Blue, Green, Red}]
   ],
   PlotLabel → "α= " <> ToString[α] <> "   Ta= " <> ToString[Td],
   PlotRange → {{0, 5}, {0, 2}}
  ]
 ]
```

```
In[ ]:= plt1[idxT_, idxr_, ss_, nn_] := Block[
   {rec = recpersite * idxr * winDist,
    M = Floor[2 nn 2 ss], Td = sweepTimes[[idxT]] / (2 * datNe), α},
   α = rec / ss Log[2 nn ss];
   Show[
    Plot[Evaluate[yuleCDFS /. {s → ss, r → rec, Ne → nn, M → M, Ta → Td} /. Peln → PELN],
     {t, 0, 6}, PlotStyle → Evaluate[Darker[#] & /@ {Blue, Green, Red}],
     Exclusions → None, PlotRange → Full],
    Plot[Evaluate[#[α, Td, t] & /@ selCdfs], {t, 0, 6}, PlotRange → Full, PlotStyle →
      Evaluate[{Darker[#], Dashed} & /@ {Blue, Green, Red}], Exclusions → None],
    Histogram[classicSweepData[[idxT]][[idxr]][[2]], {.05}, "CDF",
     ChartStyle → Evaluate[Lighter[#] & /@ {Blue, Green, Red}]
    ],
    PlotLabel → "α= " <> ToString[α] <> "   Ta= " <> ToString[Td],
    PlotRange → {{0, 5}, {0, 1.01}}
   ]
  ]
```

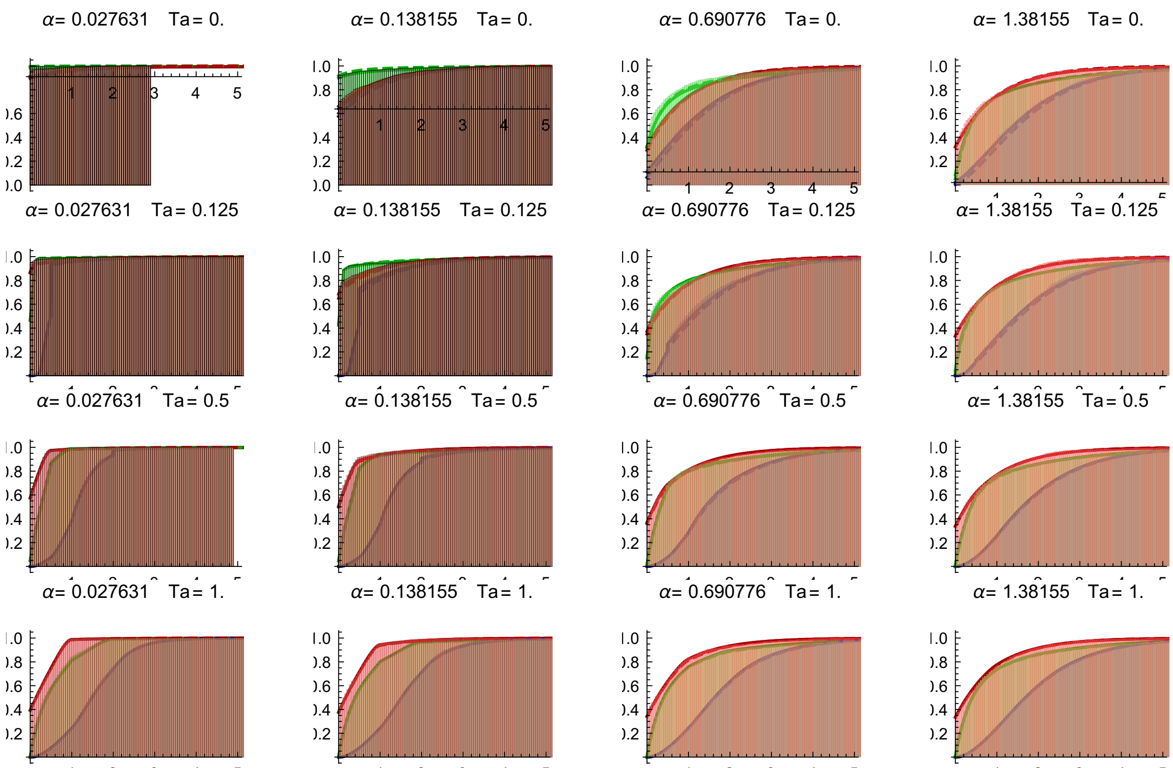
In[•]:=
```
GraphicsGrid[
 Table[
  Table[
   plt0[idxT, idxr, .05, 10 000], {idxr, {2, 10, 50, 100}}], {idxT, {1, 3, 5, 7}}],
 ImageSize → Full, Spacings → 0
]
```

Out[•]=



In[ ]:=
```
GraphicsGrid[
 Table[
  Table[
   plt1[idxT, idxr, .05, 10 000], {idxr, {2, 10, 50, 100}}], {idxT, {1, 3, 5, 7}}],
 ImageSize → Full, Spacings → 0
]
```

Out[ ]=



Site Frequency Spectrum, n = 9

```
(*n=9;
gfList = Block[{sample = Table[{1},{i,1,n}],gfDeltaEpsilon },
  gfDeltaEpsilon  =GFstar[α,ω,sample,δ] //Expand;
  gfDeltaEpsilon [[0]] = List;
  gfDeltaEpsilon  = SubSimplifyPeeRulesv2 [#,n]&/@gfDeltaEpsilon //Parallelize ;
  gfDeltaEpsilon ];

meansList  = MakeMeanList [gfList ,n,ω,t,δ,Ta];
DumpSave["iTonBL_9_gfList_meanList .mx",{gfStarList ,meansList }]*)
```

In[ ]:=
```
SetDirectory[NotebookDirectory[]];
(*DumpSave["iTonBL_9_gfList_meanList.mx",{gfStarList,meansList}]*)
Get["iTonBL_9_gfList_meanList.mx"];
meanItonFunList = Function[{α, Ta}, Evaluate[# //. P → PknToAlpha]] & /@ meanList;
sfs[α_, Ta_] := Block[{a = #[α, Ta] & /@ meanItonFunList}, a / Total[a]]
```

In[ ]:=
```
talist = {0.00000001, .1, .25, .5, 1, 1.5, 2};
alphaList = {0, 0.25, 0.5};
legendItems = "Ta= " <> ToString[#] & /@ {0, .1, .25, .5, 1, 1.5, 2};
titleItems = "α= " <> ToString[#] & /@ {0, .25, .75}

(*plt1 = getSFS[expectedBLs,0,#]&/@tslist;
plt2 = getSFS[expectedBLs,0.25,#]&/@tslist;
plt3 = getSFS[expectedBLs,0.75,#]&/@tslist;*)
plt1 = sfs[0, #] & /@ talist;
plt2 = sfs[0.25, #] & /@ talist;
plt3 = sfs[0.75, #] & /@ talist;
```

Out[ ]= {α= 0, α= 0.25, α= 0.75}

Out[ ]=