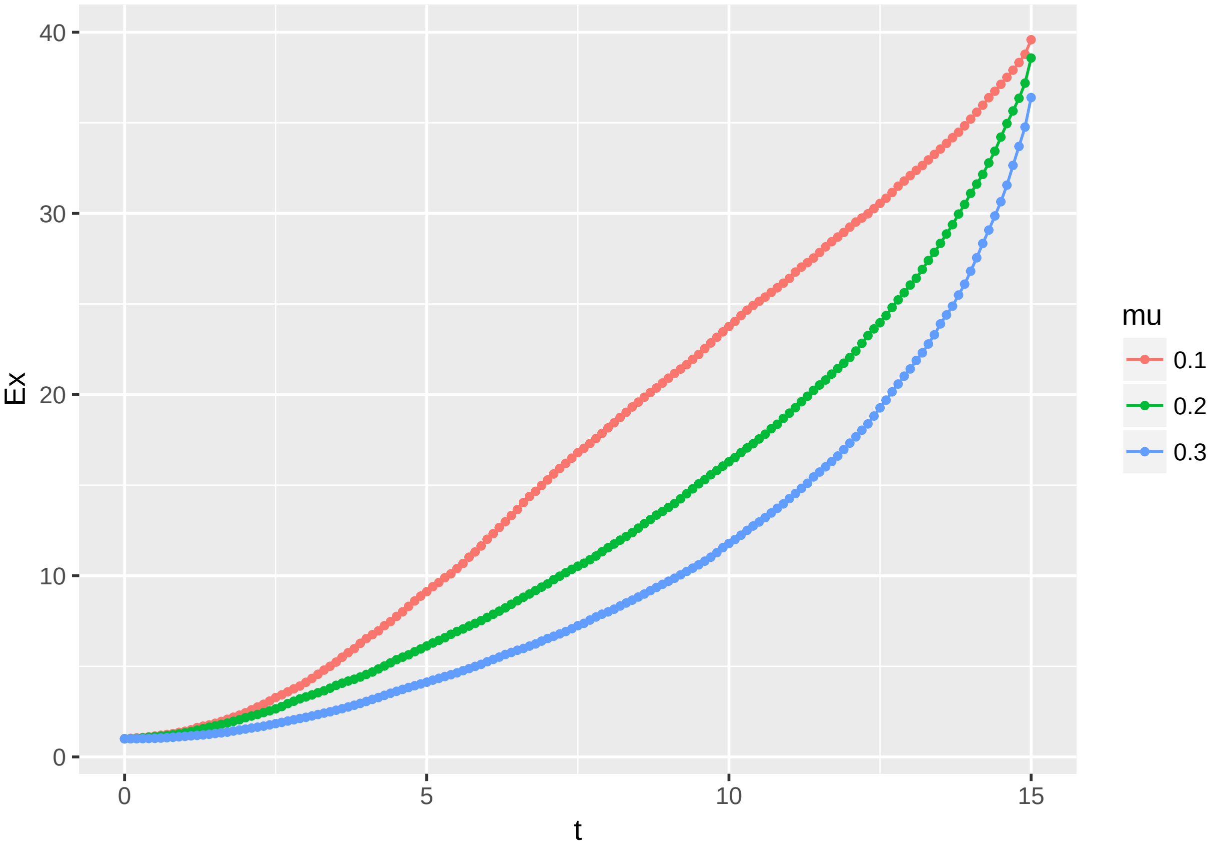
What extant species can tell us about extinction?

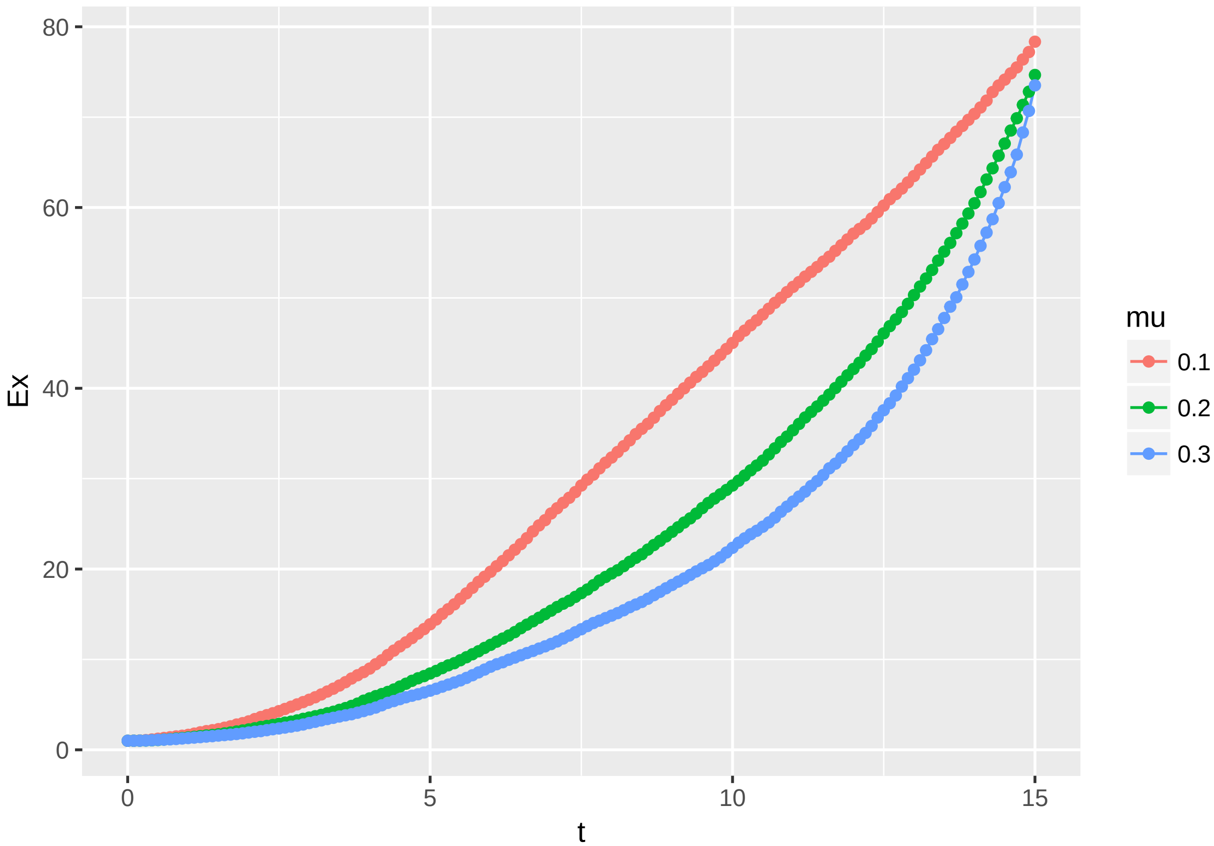
Because extinct species are rarelly included on phylogenetic trees, we are interested on invetigate the information that extant species contains about extinction rates. On the plot below we can see the expected Ltt plot, of extant-species only trees, for 3 different extinction rates



we can see a clear difference on Ltt plots of extant species, smaller extinction rates tents to grow faster on the begining whereas higger extinction rate seems to have a slow grow on the begining.

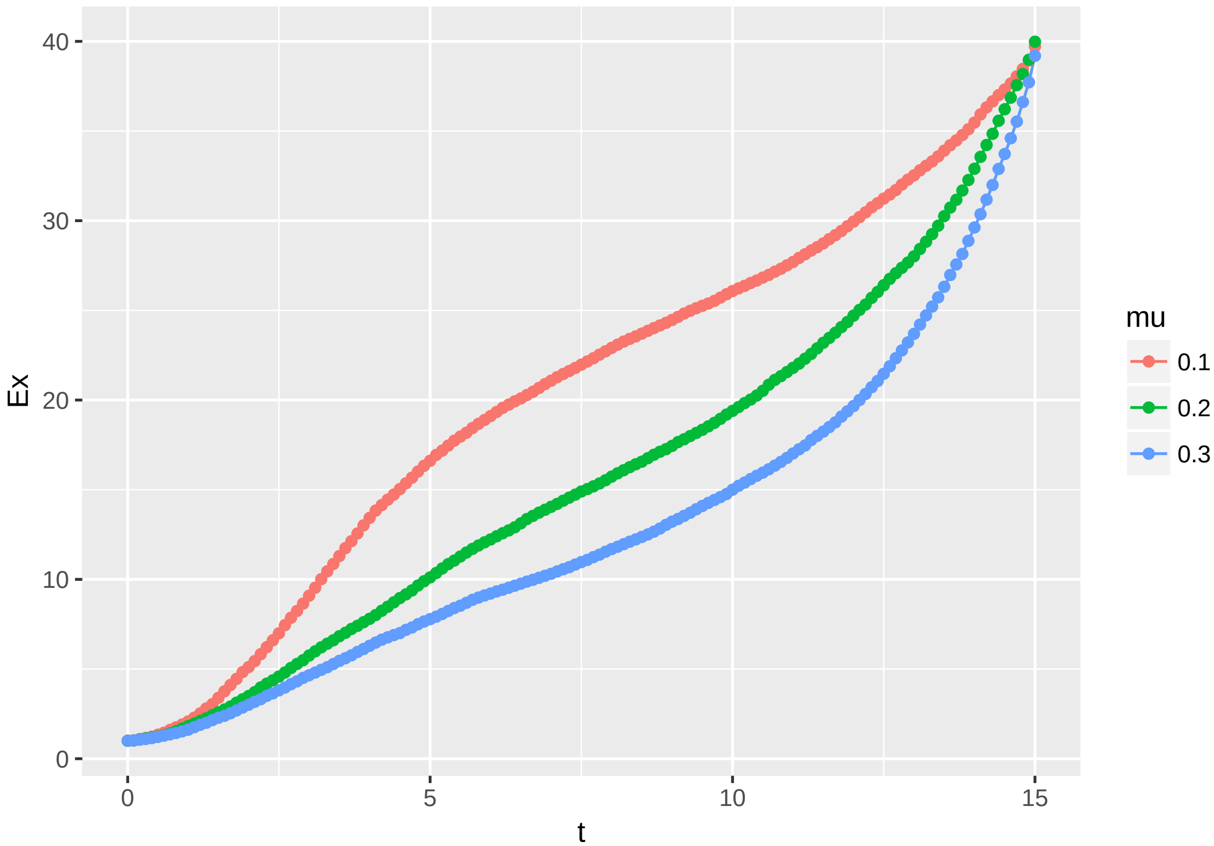
It seems also that is a matter of the first derivate, we can look at that also

Now we do it again, but with $K = 80$ rather than $K = 40$ in order to check some influence on the K parameter

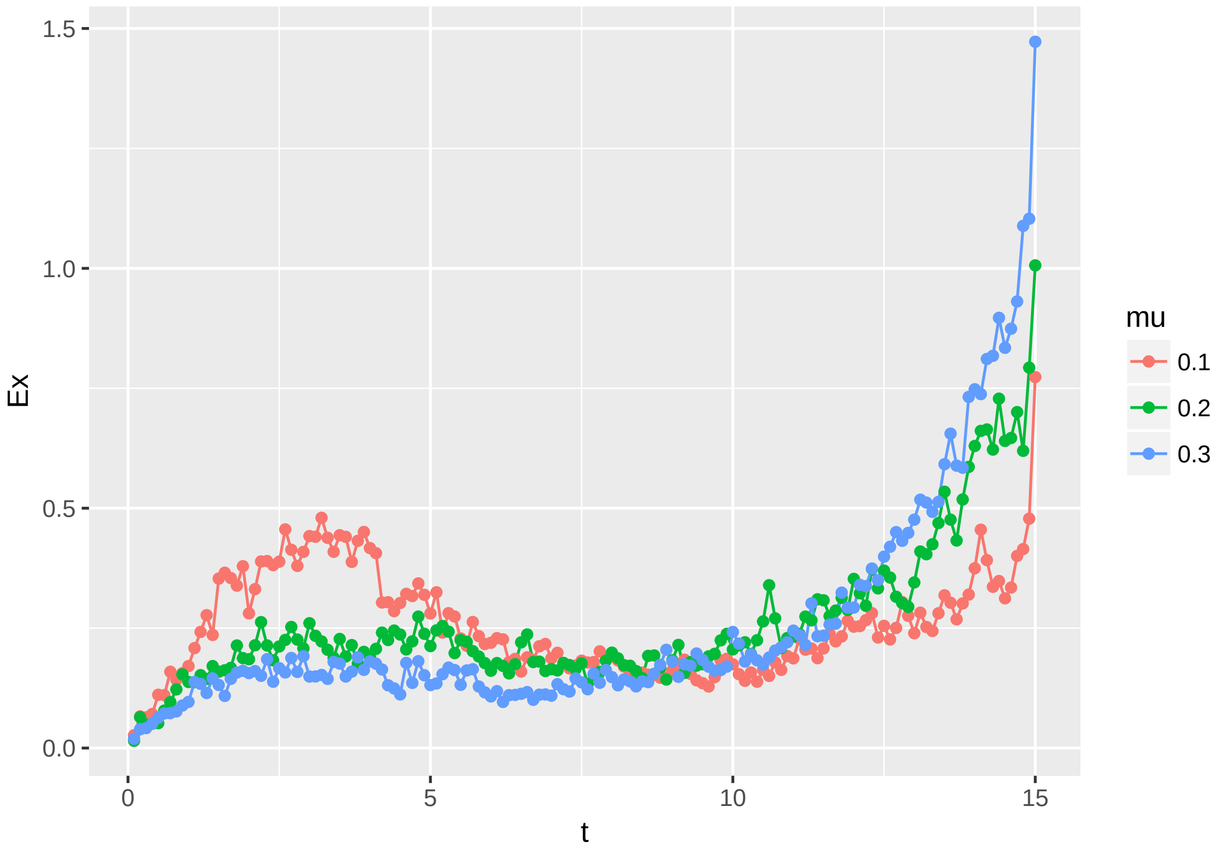


It seems it is just a change on scale. Now, what about λ?

We set $= \lambda = 1.2$ rather than $\lambda = 0.8$ and we see again the ltt plot

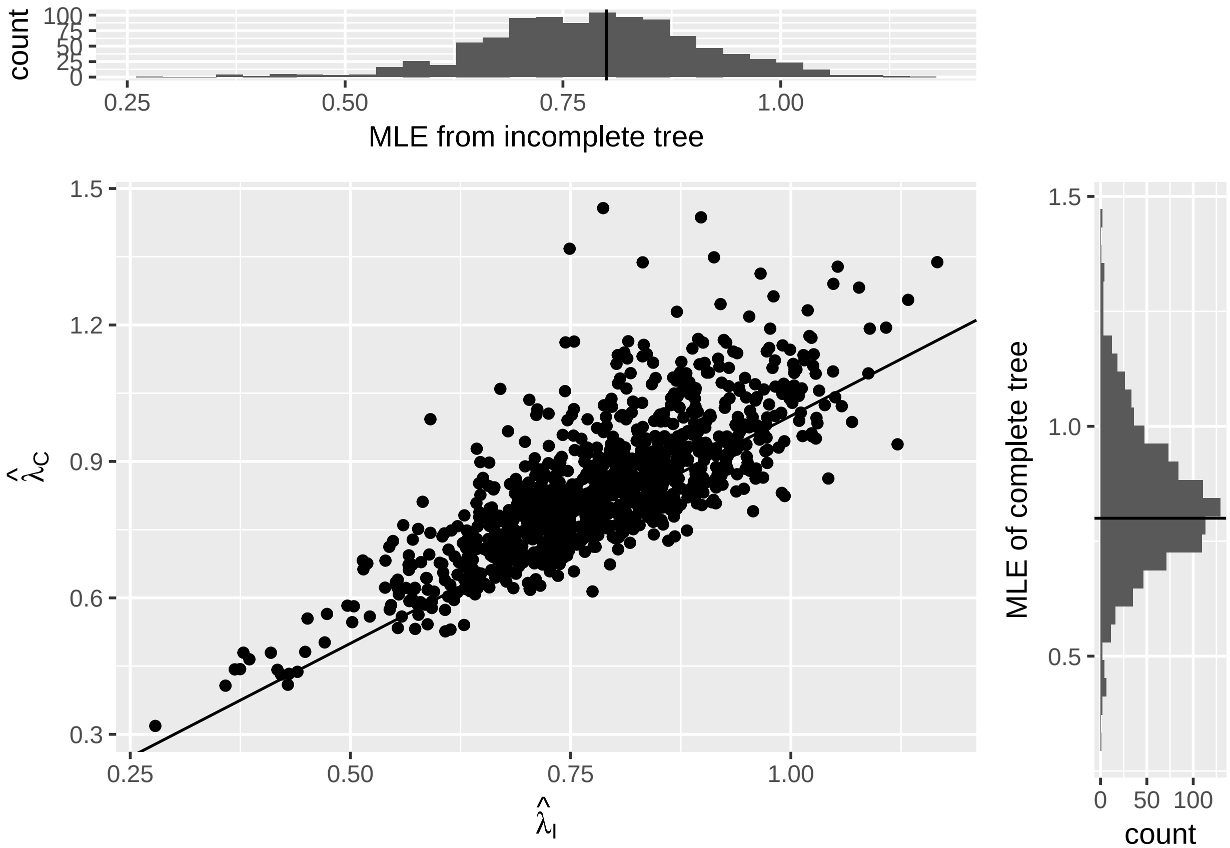


and we check the derivative again

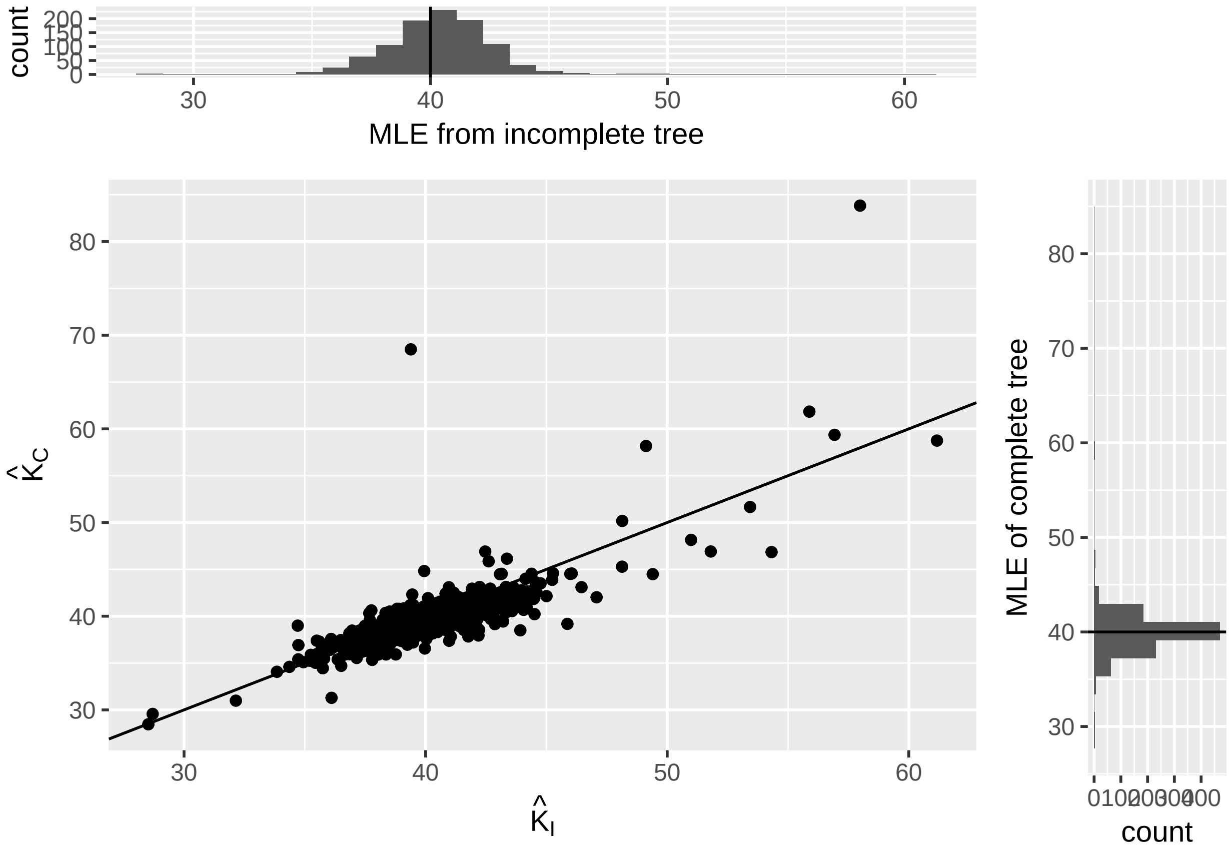


2 parameter estimation (fixing μ)

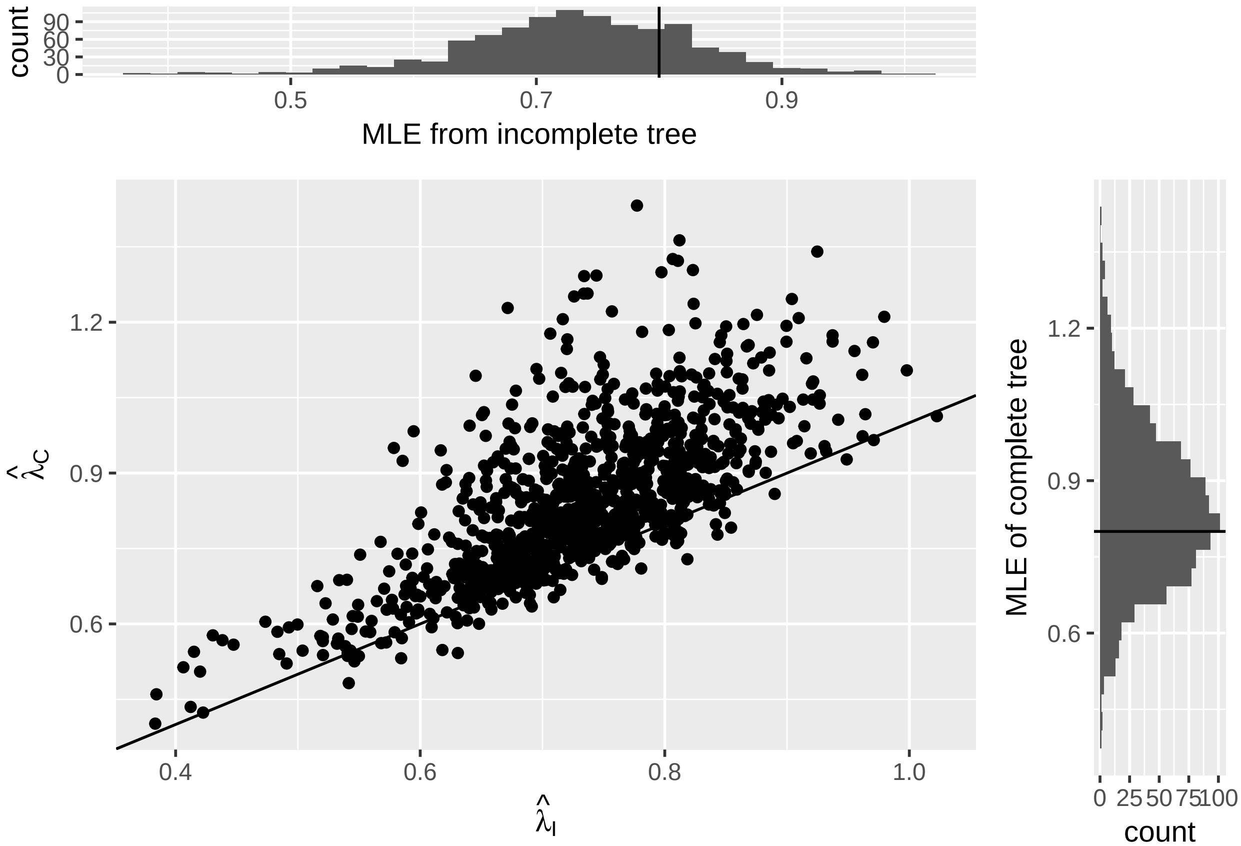
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



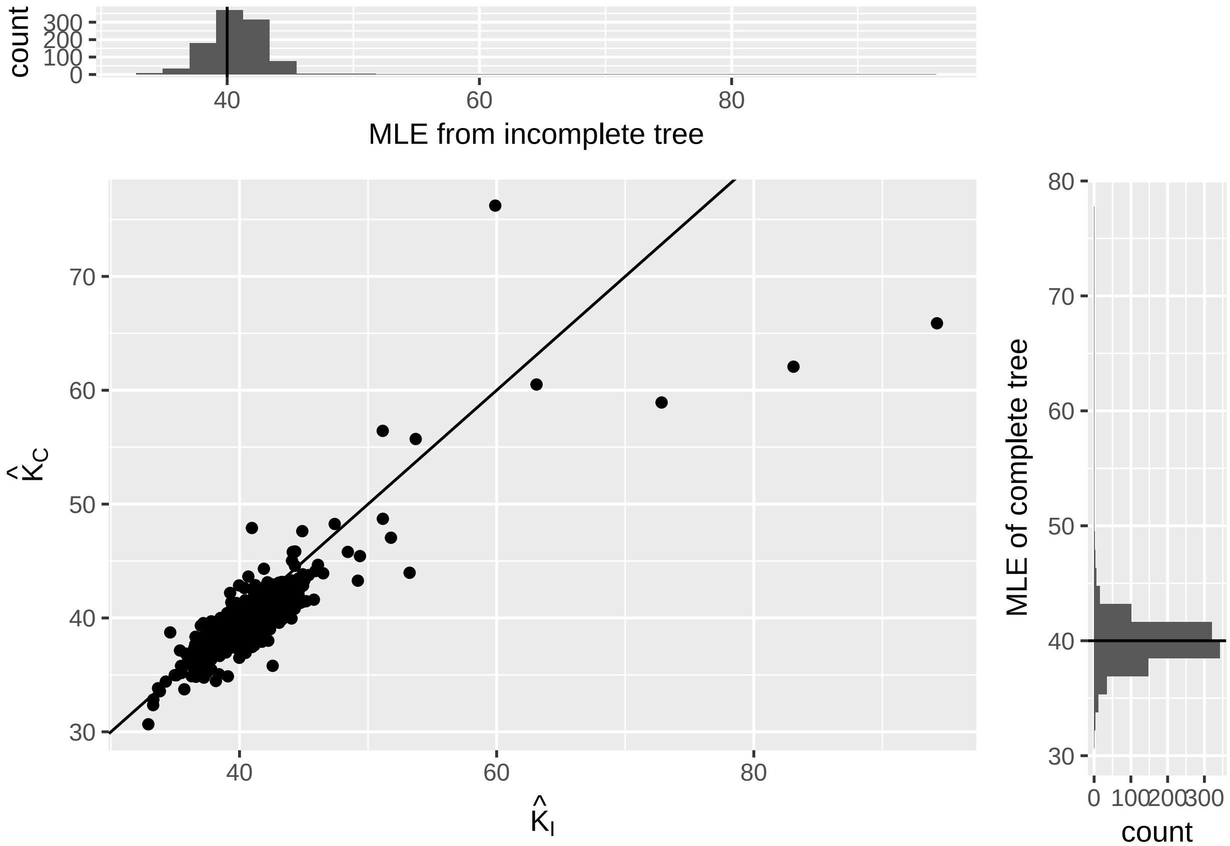

```
## [1] "0.00700000000000001 proportion of data was excluded for vizualization purposes"

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## Warning in proc.time() - p: longer object length is not a multiple of
## shorter object length

##       user     system    elapsed
## 366.612742  -0.148922 330.979588

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




```
## [1] "0.005 proportion of data was excluded for vizualization purposes"

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

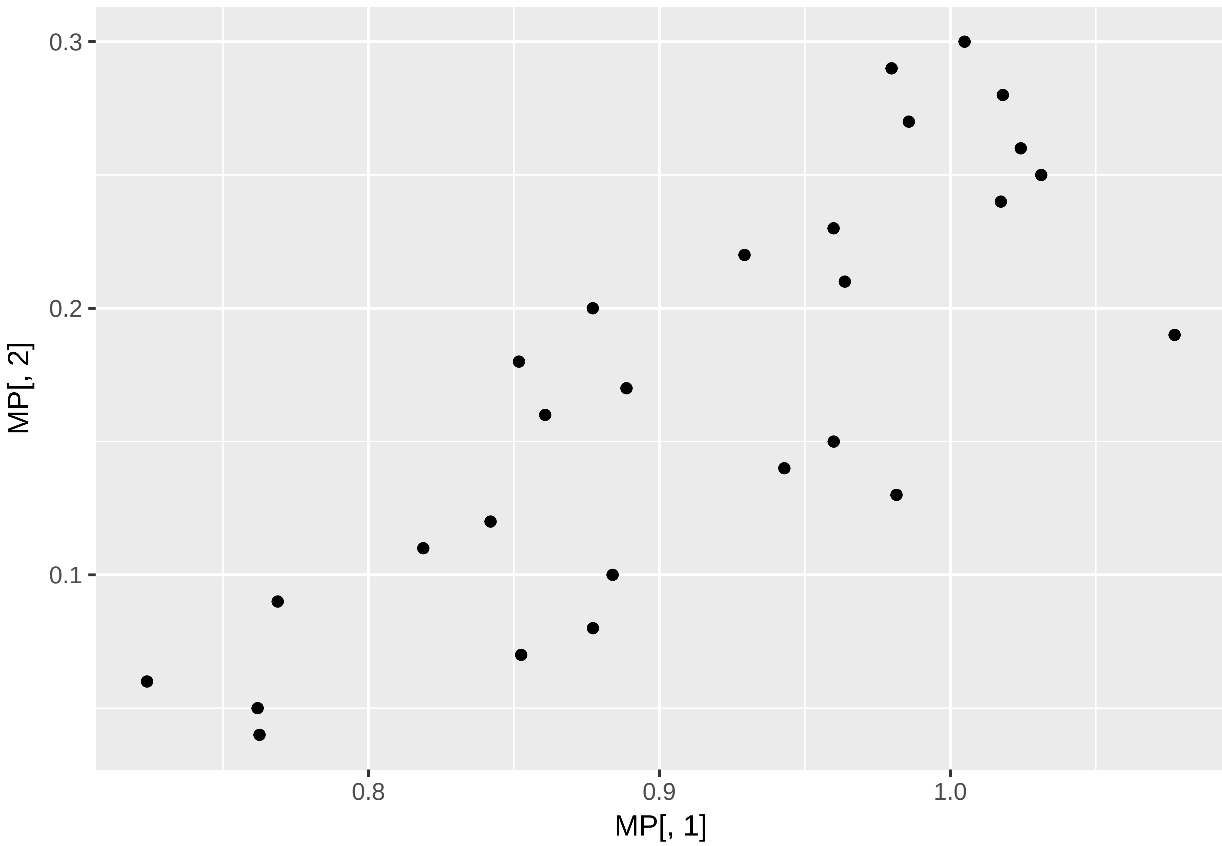



```
## Warning in proc.time() - p: longer object length is not a multiple of
## shorter object length

##          user        system       elapsed
## 2244.75775482   -0.09718191 2206.84986884
```

Ok. The estimations are fine.

Now let´s try for a grid of μ

```
mu0 = seq(0.04,0.3,by=0.01)
s = sim_phyl(seed=3)
p <- subplex(par = c(2,0.2,60), fn = llik, n = s$n, E = s$E, t = s$wt)$par
wt = (s$newick.extant.p)$wt
MP = matrix(nrow=length(mu0), ncol=3)
n_trees = 10
for(i in 1:length(mu0)){
  mu = mu0[i]
  trees = sim_srt(wt=wt, pars=c(p[1],mu,p[3]), parallel = F, n_trees = n_trees)
  pars = subplex(par = c(2,60), fn = llik_st , setoftrees = trees, mu = mu, impsam = FALSE)$par
  MP[i,] = c(pars[1],mu,pars[2])
}
qplot(MP[,1],MP[,2])
```



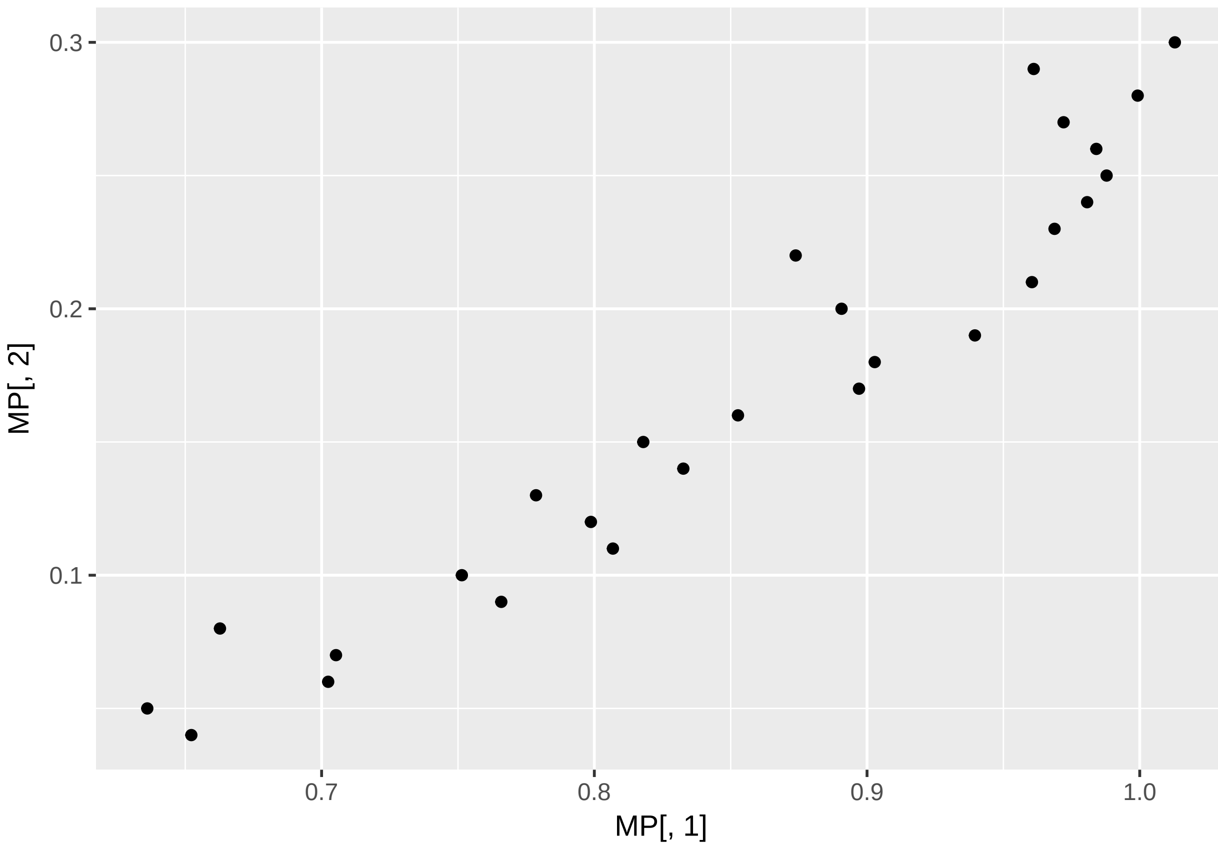

```
lm1 = lm(MP[,2] ~ MP[,1])
summary(lm1)
```

```
##
## Call:
## lm(formula = MP[, 2] ~ MP[, 1])
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.090752 -0.039528  0.005664  0.033584  0.074825
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.44537    0.08967  -4.967 4.06e-05 ***
## MP[, 1]      0.67416    0.09773   6.898 3.13e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0475 on 25 degrees of freedom
## Multiple R-squared:  0.6556, Adjusted R-squared:  0.6418
## F-statistic: 47.59 on 1 and 25 DF,  p-value: 3.131e-07
```

```
l1 = MP[,1]
m1 = MP[,2]
k1 = MP[,3]
MP1=MP
```

Does it help 100 trees (probably not)

```
mu0 = seq(0.04,0.3,by=0.01)
s = sim_phyl(seed=3)
p <- subplex(par = c(2,0.2,60), fn = llik, n = s$n, E = s$E, t = s$wt)$par
wt = (s$newick.extant.p)$wt
MP = matrix(nrow=length(mu0), ncol=3)
n_trees = 100
for(i in 1:length(mu0)){
  mu = mu0[i]
  trees = sim_srt(wt=wt, pars=c(p[1],mu,p[3]), parallel = F, n_trees = n_trees)
  pars = subplex(par = c(2,60), fn = llik_st , setoftrees = trees, mu = mu, impsam = FALSE)$par
  MP[i,] = c(pars[1],mu,pars[2])
}
qplot(MP[,1],MP[,2])
```




```
lm2 = lm(MP[,2] ~ MP[,1])
summary(lm2)
```

```
##
## Call:
## lm(formula = MP[, 2] ~ MP[, 1])
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.034392 -0.014381 -0.003059  0.013309  0.051681
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

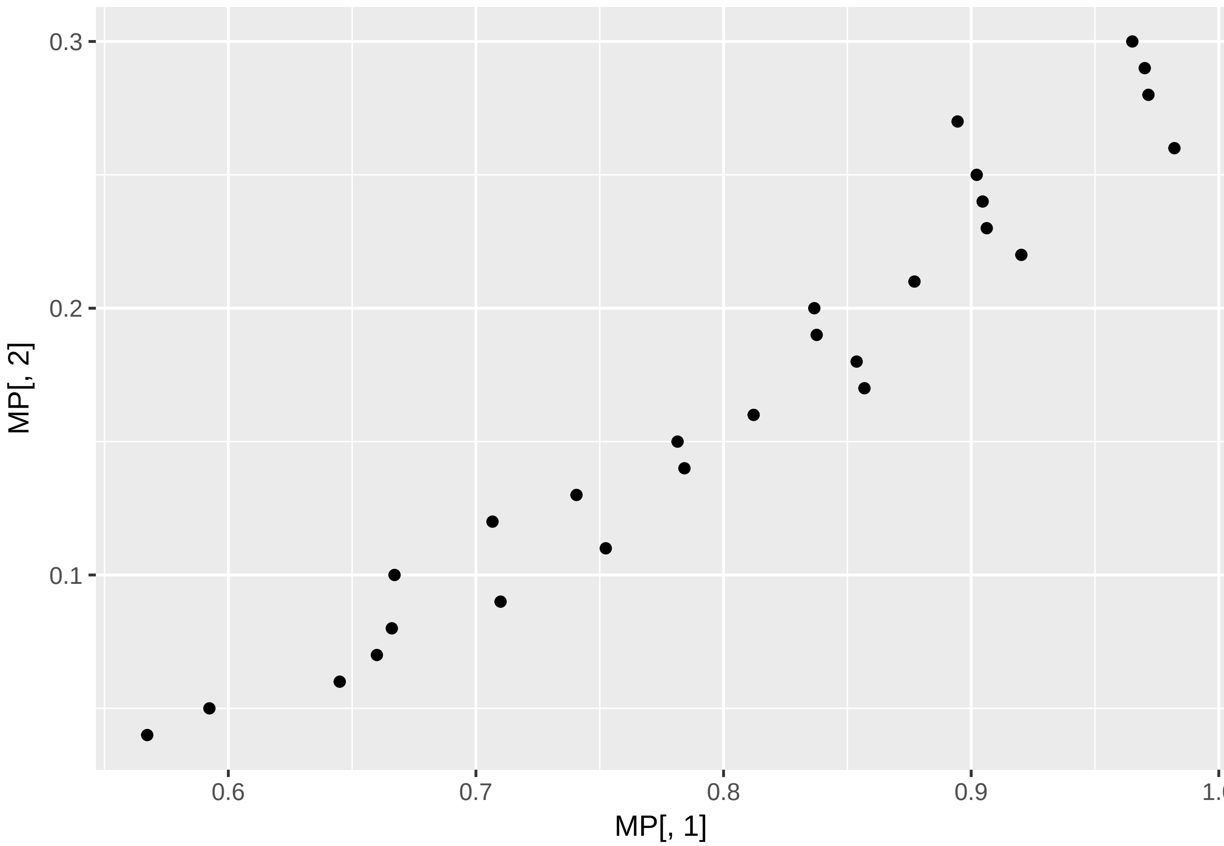
```
## (Intercept) -0.38250    0.03282  -11.65 1.34e-11 ***
## MP[, 1]      0.64593    0.03802   16.99 3.05e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02285 on 25 degrees of freedom
## Multiple R-squared:  0.9203, Adjusted R-squared:  0.9171
## F-statistic: 288.6 on 1 and 25 DF,  p-value: 3.053e-15
```

```
l2 = MP[,1]
m2 = MP[,2]
k2 = MP[,3]
MP2=MP
```

Actually, the variance decreases.

1000 trees?

```
mu0 = seq(0.04,0.3,by=0.01)
s = sim_phyl(seed=3)
p <- subplex(par = c(2,0.2,60), fn = llik, n = s$n, E = s$E, t = s$wt)$par
wt = (s$newick.extant.p)$wt
MP = matrix(nrow=length(mu0), ncol=3)
n_trees = 1000
for(i in 1:length(mu0)){
  mu = mu0[i]
  trees = sim_srt(wt=wt, pars=c(p[1],mu,p[3]), parallel = F, n_trees = n_trees)
  pars = subplex(par = c(2,60), fn = llik_st , setoftrees = trees, mu = mu, impsam = FALSE)$par
  MP[i,] = c(pars[1],mu,pars[2])
}
qplot(MP[,1],MP[,2])
```



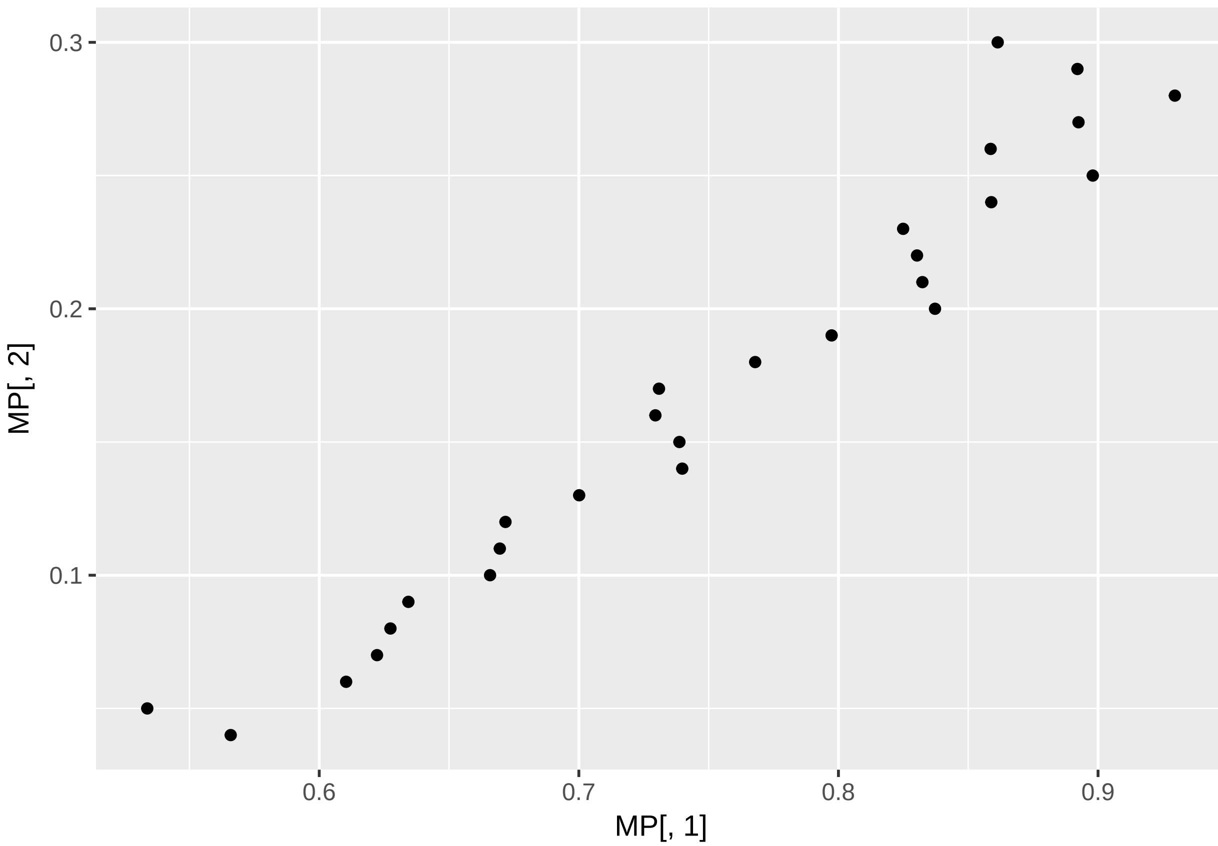

```
lm3 = lm(MP[,2] ~ MP[,1])
summary(lm3)
```

```
##
## Call:
## lm(formula = MP[, 2] ~ MP[, 1])
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.032094 -0.015010 -0.001572  0.013852  0.044157
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.33899    0.02463  -13.76  3.6e-13 ***
## MP[, 1]      0.63144    0.03022   20.89  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01884 on 25 degrees of freedom
## Multiple R-squared:  0.9458, Adjusted R-squared:  0.9437
## F-statistic: 436.6 on 1 and 25 DF,  p-value: < 2.2e-16
```

```
l3 = MP[,1]
m3 = MP[,2]
```

10000 trees?

```r
mu0 = seq(0.04,0.3,by=0.01)
s = sim_phyl(seed=3)
p <- subplex(par = c(2,0.2,60), fn = llik, n = s$n, E = s$E, t = s$wt)$par
wt = (s$newick.extant.p)$wt
MP = matrix(nrow=length(mu0), ncol=3)
n_trees = 10000
for(i in 1:length(mu0)){
  mu = mu0[i]
  trees = sim_srt(wt=wt, pars=c(p[1],mu,p[3]), parallel = F, n_trees = n_trees)
  pars = subplex(par = c(2,60), fn = llik_st , setoftrees = trees, mu = mu, impsam = FALSE)$par
  MP[i,] = c(pars[1],mu,pars[2])
}
qplot(MP[,1],MP[,2])
```




```r
lm4 = lm(MP[,2] ~ MP[,1])
summary(lm4)
```

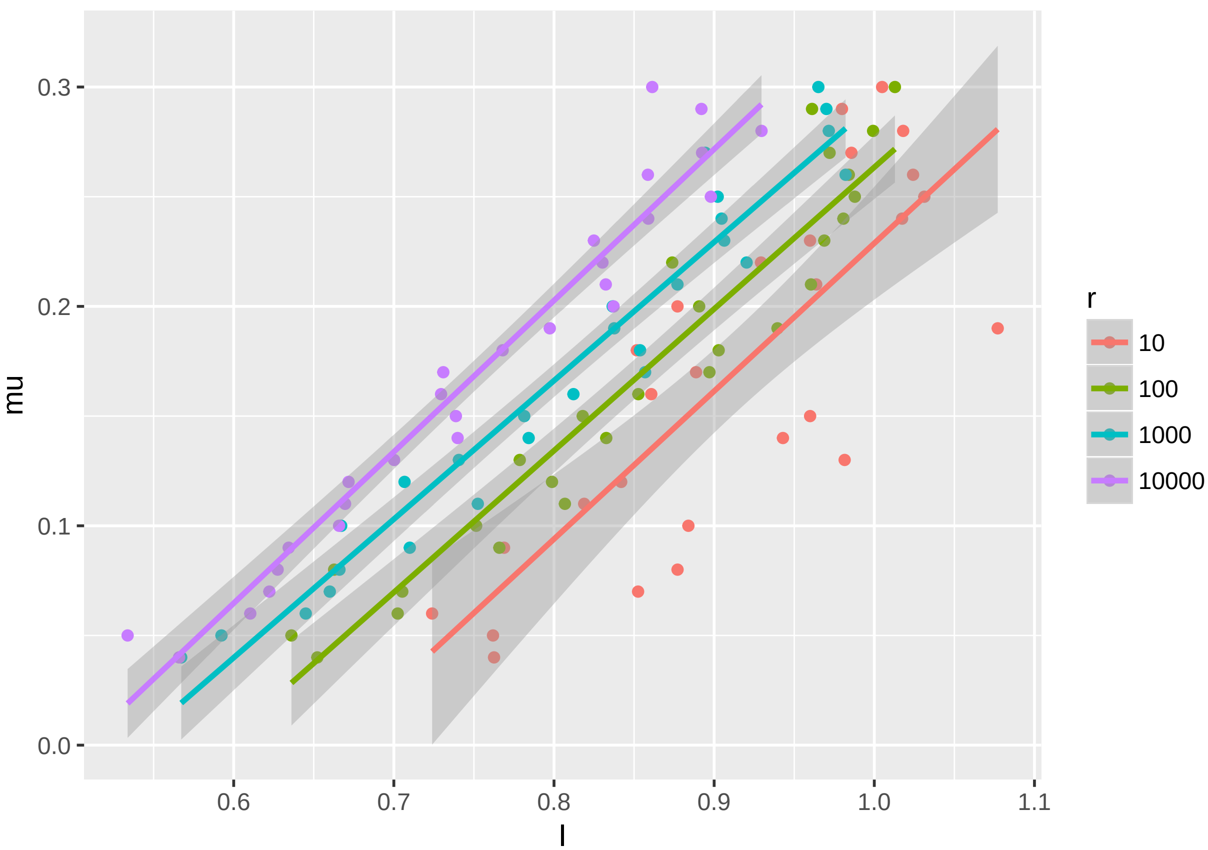
```
##
## Call:
## lm(formula = MP[, 2] ~ MP[, 1])
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.028335 -0.010639 -0.003290  0.005892  0.055001
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) -0.34923    0.02364  -14.77 7.43e-14 ***
## MP[, 1]      0.68988    0.03108   22.19  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01779 on 25 degrees of freedom
## Multiple R-squared:  0.9517, Adjusted R-squared:  0.9498
## F-statistic: 492.6 on 1 and 25 DF,  p-value: < 2.2e-16
```

```r
l4 = MP[,1]
m4 = MP[,2]
```

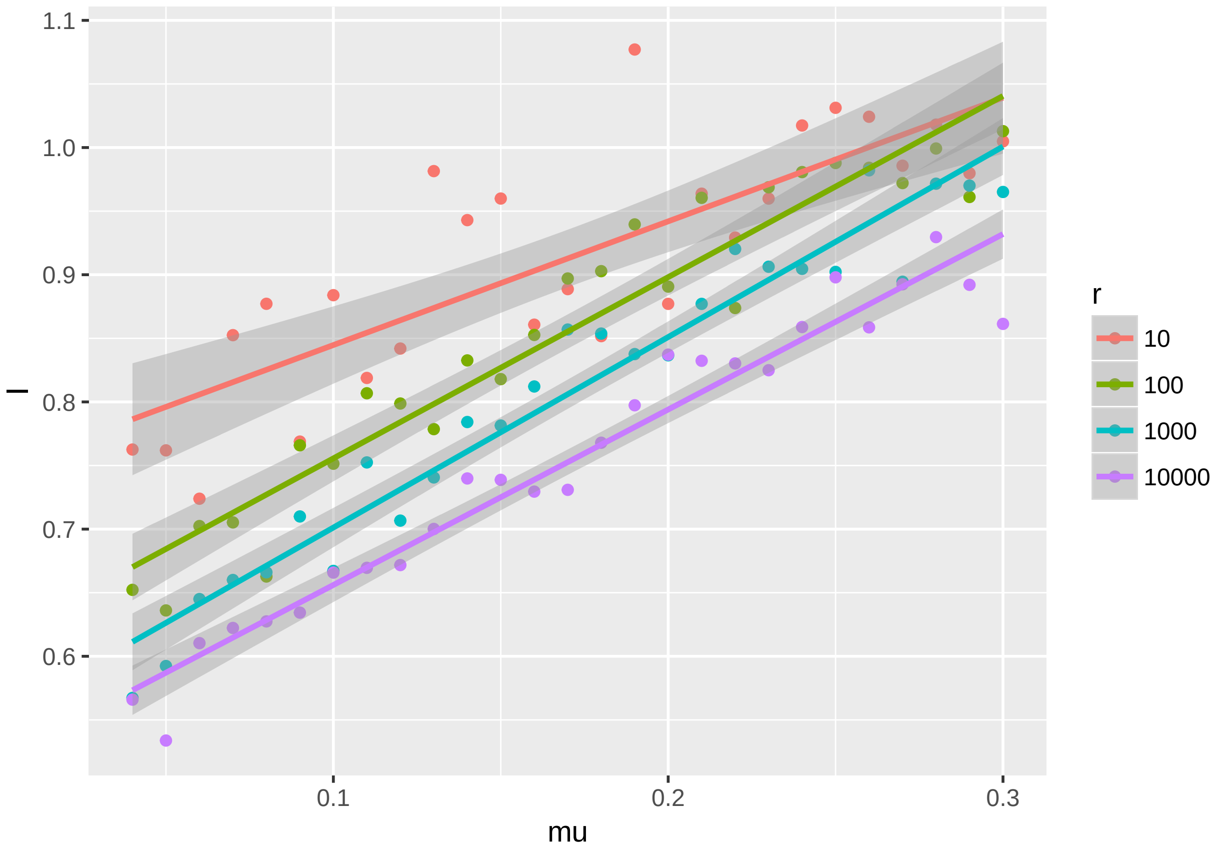
let's look at them all together

```r
regressions = data.frame(mu = rep(m1,4), l = c(l1,l2,l3,l4), r = c(rep(x = 10,length(m1)),rep(x = 100,le
regressions$r = as.factor(regressions$r)
ggplot(data=regressions, aes(x=l,y=mu,colour=r)) + geom_point() + geom_smooth(method='lm')
```



Just for convenience: μ vs λ

```r
ggplot(data=regressions, aes(x=mu,y=l,colour=r)) + geom_point() + geom_smooth(method='lm')
```



get coefficients

```
lm11 = lm(l1~m1)
lm22 = lm(l2~m2)
lm33 = lm(l3~m3)
lm44 = lm(l4~m4)
lm11$coef
```

```
## (Intercept)          m1
##    0.747476    0.972451
```

```
lm22$coef
```

```
## (Intercept)          m2
##   0.6131469   1.4247474
```

```
lm33$coef
```

```
## (Intercept)          m3
##   0.5514301   1.4979030
```

```
lm44$coef
```

```
## (Intercept)          m4
##   0.5181201   1.3795075
```

```
lm11$coef[1] + lm11$coef[2]*0.1
```

```
## (Intercept)
##   0.8447211
```

```r
lm22$coef[1] + lm22$coef[2]*0.1
```

```
## (Intercept)
##   0.7556216
```

```r
lm33$coef[1] + lm33$coef[2]*0.1
```

```
## (Intercept)
##   0.7012204
```

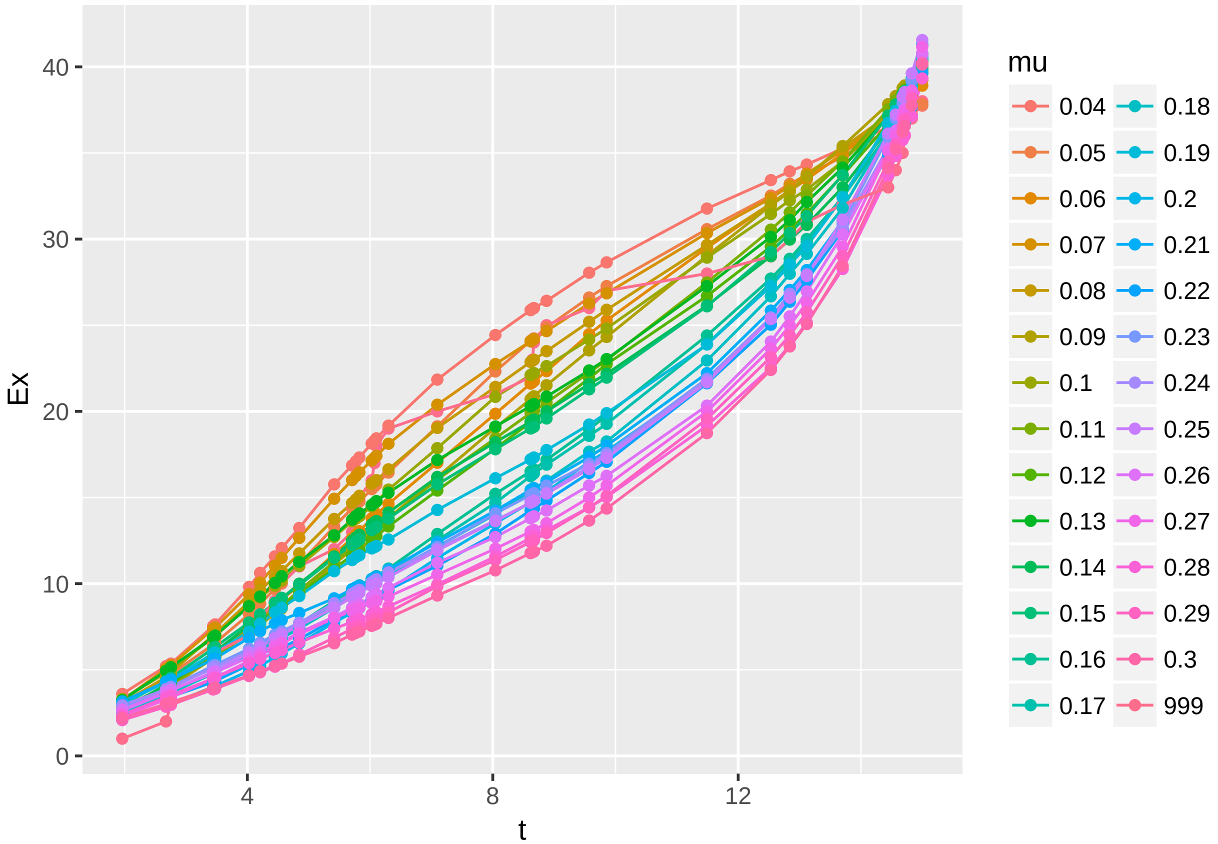
```r
lm44$coef[1] + lm44$coef[2]*0.1
```

```
## (Intercept)
##   0.6560708
```

The Ltt plot

Now let's try to minimize ltt, but first vizualize it

```
time = proc.time()
grid = wt
Ltt = data.frame(t=cumsum(wt), Ex = (s$newick.extant.p)$n, mu=999)
for(i in 1:length(mu0)){
  mu = mu0[i]
  pars = c(l1[i],m1[i],k1[i])
  ltt = data.frame(t = cumsum(wt), Ex = expectedLTT(pars,drop.extinct = TRUE, grid=cumsum(wt))$Ex, mu=mu
  Ltt = rbind(Ltt,ltt)
}
Ltt$mu = as.factor(Ltt$mu)
ggplot(data=Ltt, aes(x=t, y=Ex, colour = mu)) + geom_line() +   geom_point()
```




```
print(proc.time()-time)
```

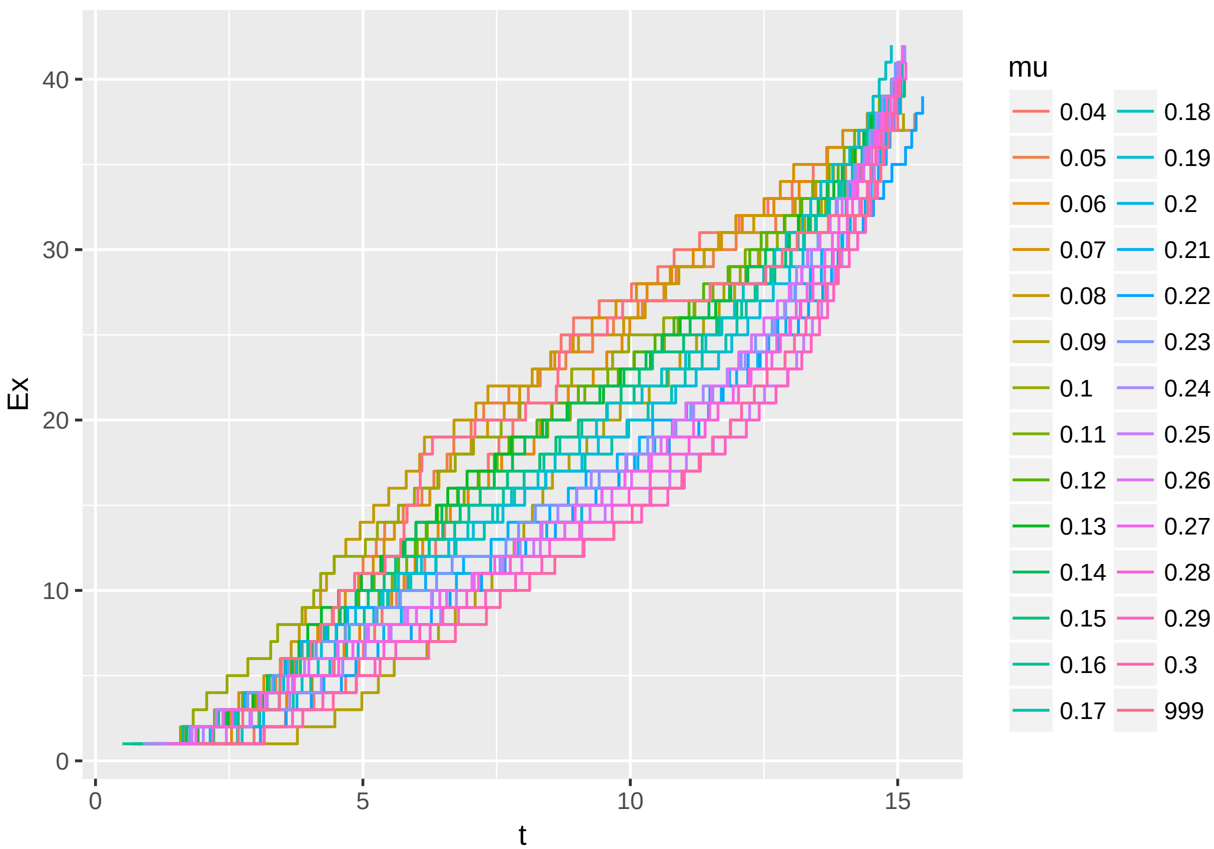
```
##    user  system elapsed
## 800.928   0.028 801.029
```

```
time = proc.time()
Ltt2 = data.frame(t=cumsum(wt), Ex = (s$newick.extant.p)$n, mu=999)
for(i in 1:length(mu0)){
  mu = mu0[i]
  pars = c(l1[i],m1[i],k1[i])
  expe = expectedLTT2(pars)
  ltt2 = data.frame(t = cumsum(expe$t), Ex = expe$Ex, mu=mu)
```

```
  Ltt2 = rbind(Ltt2,ltt2)
}
Ltt2$mu = as.factor(Ltt2$mu)
print(proc.time()-time)
```

```
##    user  system elapsed
##  84.236   0.028  84.286
```

```
ggplot(data=Ltt2, aes(x=t, colour = mu)) + geom_step(aes(y=Ex)) #+  scale_color_brewer(name = "Method",
```




```
ltt1 = Ltt[Ltt$mu == 999,]
diff_ltt = NaN
sq_diff = NaN
diff_mat = matrix(0,ncol=length(mu0),nrow=length(wt))
for(i in 1:length(mu0)){
 mu = mu0[i]
 ltt = Ltt[Ltt$mu == mu,]
 ltt$Ex = abs(ltt1$Ex-ltt$Ex)
 diff_mat[,i] = ltt$Ex
 diff_ltt[i] = sum(ltt$Ex)
 sq_diff[i] = sum((ltt1$Ex-ltt$Ex)^2)

}
n_diff_ltt = diff_ltt*wt
```

```
## Warning in diff_ltt * wt: longer object length is not a multiple of shorter
## object length
```

```r
n_diff_ltt
```

```
##  [1] 181.975661  38.790611   5.465826  49.950100   1.084340   1.444825
##  [7]  37.970172  16.160830  21.491267   8.775069  22.755875  52.655763
## [13]  39.821809   8.955118   8.924846  23.931219   7.819554   4.824470
## [19]  34.211471 129.433928 152.438611  97.228231   5.400489   4.735825
## [25]  41.375117 146.610782  62.935686 151.773133  56.694765  20.956833
## [31]  20.161568  37.817549  66.199259   8.516344   9.937103   3.252853
## [37]   9.035809  13.466519
```

```r
diff_ltt_M = data.frame(mu = mu0, diff_ltt = diff_ltt)
choosed_mu = diff_ltt_M[diff_ltt_M$diff_ltt == min(diff_ltt_M$diff_ltt) ,]
choosed_mu$mu
```

```
## [1] 0.05
```

```r
b = matrix(0,ncol=length(mu0),nrow=length(wt))
for(i in 1:length(mu0)){
   b[i,] = diff_mat[i,]==rowMins(diff_mat)[i]
}
colSums(b)
```

```
##  [1] 2 2 3 5 3 1 3 0 0 1 0 1 0 0 0 1 2 0 0 0 0 1 0 0 0 2 0
```

```r
MP[MP[,2] == choosed_mu$mu]
```

```
## [1]  0.5337676  0.0500000 42.5976211
```

```r
MP1[max(colSums(b)) == colSums(b)]
```

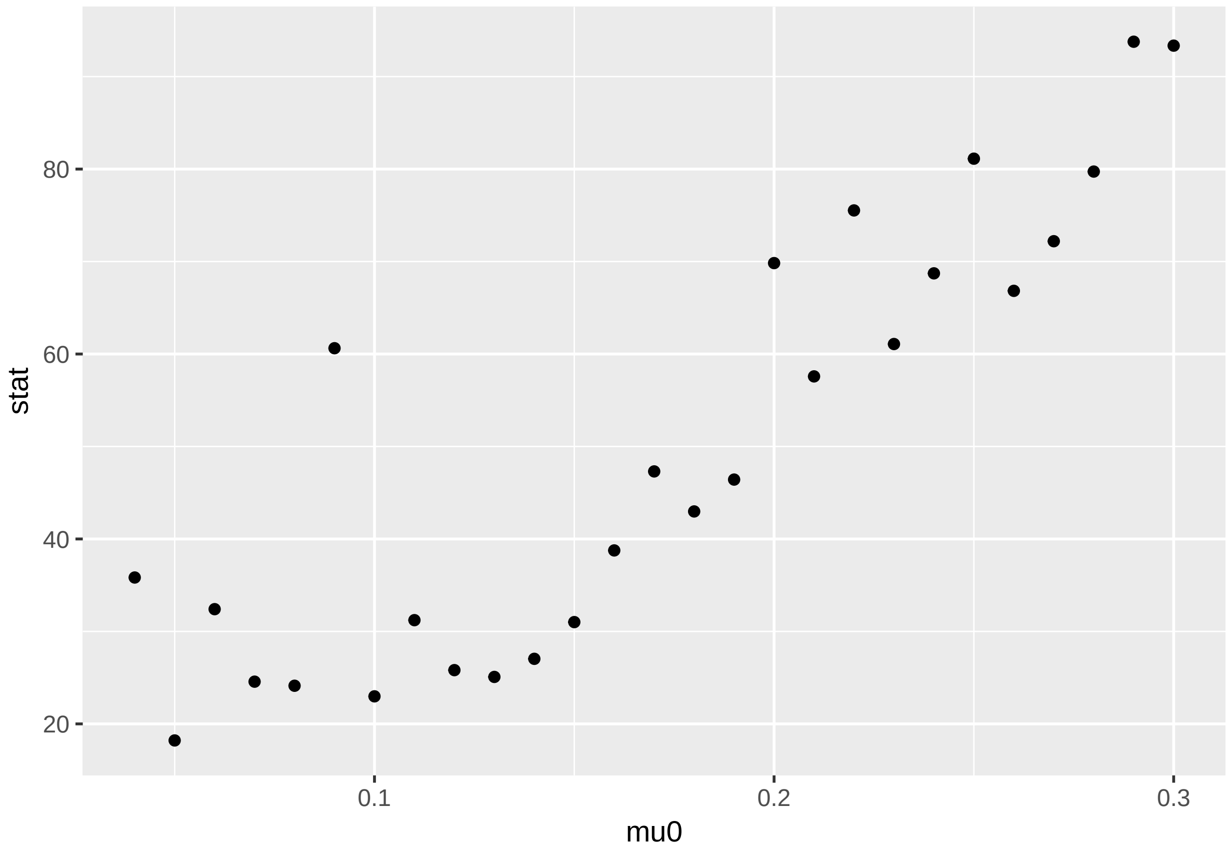
```
## [1]  0.8525022  0.0700000 39.4487132
```

```r
b[,max(colSums(b)) == colSums(b)]
```

```
##  [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 1 0 0 0 1 0 1 1 0 0 0 0 0 0 0 0
## [36] 0 0 0
```

Let´s use the second method

```r
stat=NULL
for(i in 1:length(mu0)){
  ltt_temp = Ltt2[Ltt2$mu == mu0[i],]
  ltt_temp = rbind(ltt_temp, data.frame(t=15,Ex=ltt_temp$Ex[length(ltt_temp$Ex)]+1,mu=ltt_temp$mu[1]))
  pp = list(wt=c(ltt_temp$t[1],diff(ltt_temp$t)),E=rep(1,(length(ltt_temp$t)-1)),n=ltt_temp$Ex)
  phy = p2phylo(pp)
  stat[i] = ltt_stat(s$newick.extant,phy)
}

qplot(mu0,stat)
```



it looks fine, but 10 trees has too much variation, let´s repeat it for 100 trees

```
#ct = 15
#dt = 0.1
#grid = wt
Ltt = data.frame(t=cumsum(wt), Ex = (s$newick.extant.p)$n, mu=999)

for(i in 1:length(mu0)){
  mu = mu0[i]
  pars = c(l2[i],m2[i],k2[i])
  ltt = data.frame(t = cumsum(wt), Ex = expectedLTT(pars,drop.extinct = TRUE, grid=cumsum(wt))$Ex, mu=mu
  Ltt = rbind(Ltt,ltt)
}
Ltt$mu = as.factor(Ltt$mu)
ggplot(data=Ltt, aes(x=t, y=Ex, colour = mu)) + geom_line() +   geom_point()
```

```
ltt1 = Ltt[Ltt$mu == 999,]
diff_ltt = NaN
sq_diff = NaN
diff_mat = matrix(0,ncol=length(mu0),nrow=length(wt))
for(i in 1:length(mu0)){
 mu = mu0[i]
 ltt = Ltt[Ltt$mu == mu,]
 ltt$Ex = abs(ltt1$Ex-ltt$Ex)
 diff_mat[,i] = ltt$Ex
 diff_ltt[i] = sum(ltt$Ex)
 sq_diff[i] = sum((ltt1$Ex-ltt$Ex)^2)
```

```
}
diff_ltt_M = data.frame(mu = mu0, diff_ltt = diff_ltt)
choosed_mu = diff_ltt_M[diff_ltt_M$diff_ltt == min(diff_ltt_M$diff_ltt) ,]
choosed_mu$mu
b = matrix(0,ncol=length(mu0),nrow=length(wt))
for(i in 1:length(mu0)){
   b[i,] = diff_mat[i,]==rowMins(diff_mat)[i]
}
colSums(b)
MP[MP[,2] == choosed_mu$mu]
MP2[max(colSums(b)) == colSums(b)]
#qqplot()
```

(not working yet)

Meta Analysis

Now we are prepared to estimate parameters ofr a set of (100) trees and see the distribution.

The algorithm is:

1. simulate tree and save MLE. Drop extinct species and save ltt
2. create a grid μ_g and run monte-carlo for every $\mu \in mu_g$, then get $(\lambda(\mu), mu, K(\mu)), \forall \mu \in \mu_g$ and the corresponding ltt
3. take the best $(\lambda(\mu), \mu, K(\mu))$ taking min ltt

```
ct = 15
dt = 0.1
#grid = seq(0,ct, by=dt)
n_it = 10
mu0 = seq(0.04,0.35,by=0.03)
n_trees = 10
MMP = matrix(nrow=n_it, ncol=3)
RMP = matrix(nrow=n_it, ncol=3)
for(j in 1:n_it){
  s = sim_phyl()
  p <- subplex(par = c(2,0.2,60), fn = llik, n = s$n, E = s$E, t = s$wt)$par
  MMP[j,] = p
  s2 = s$newick.extant.p
  wt = s2$wt
  grid = cumsum(wt)
  ltt1 = data.frame(t=grid, Ex = s2$n, mu=999)
  Ltt = ltt1
  MP = matrix(nrow=length(mu0), ncol=3)
  for(i in 1:length(mu0)){
    mu = mu0[i]
    trees = sim_srt(wt=wt, pars=c(p[1],mu,p[3]), parallel = F, n_trees = n_trees)
    pars = subplex(par = c(2,60), fn = llik_st , setoftrees = trees, mu = mu, impsam = FALSE)$par
    pars = c(pars[1],mu,pars[2])
    MP[i,] = pars
    ltt = data.frame(t = grid, Ex = expectedLTT(pars,drop.extinct = TRUE, grid=grid)$Ex, mu=mu)
    Ltt = rbind(Ltt,ltt)
  }
  diff_mat = matrix(0,ncol=length(mu0),nrow=length(wt))
```

```
  for(i in 1:length(mu0)){
   mu = mu0[i]
   ltt = Ltt[Ltt$mu == mu,]
   ltt$Ex = abs(ltt1$Ex-ltt$Ex)
   diff_mat[,i] = ltt$Ex
 }
 b = matrix(0,ncol=length(mu0),nrow=length(wt))
 for(i in 1:length(mu0)){
   b[i,] = diff_mat[i,]==rowMins(diff_mat)[i]
 }
 RMP[j,] = MP[max(colSums(b)) == colSums(b),]
}
```

```
RMP
MMP
```