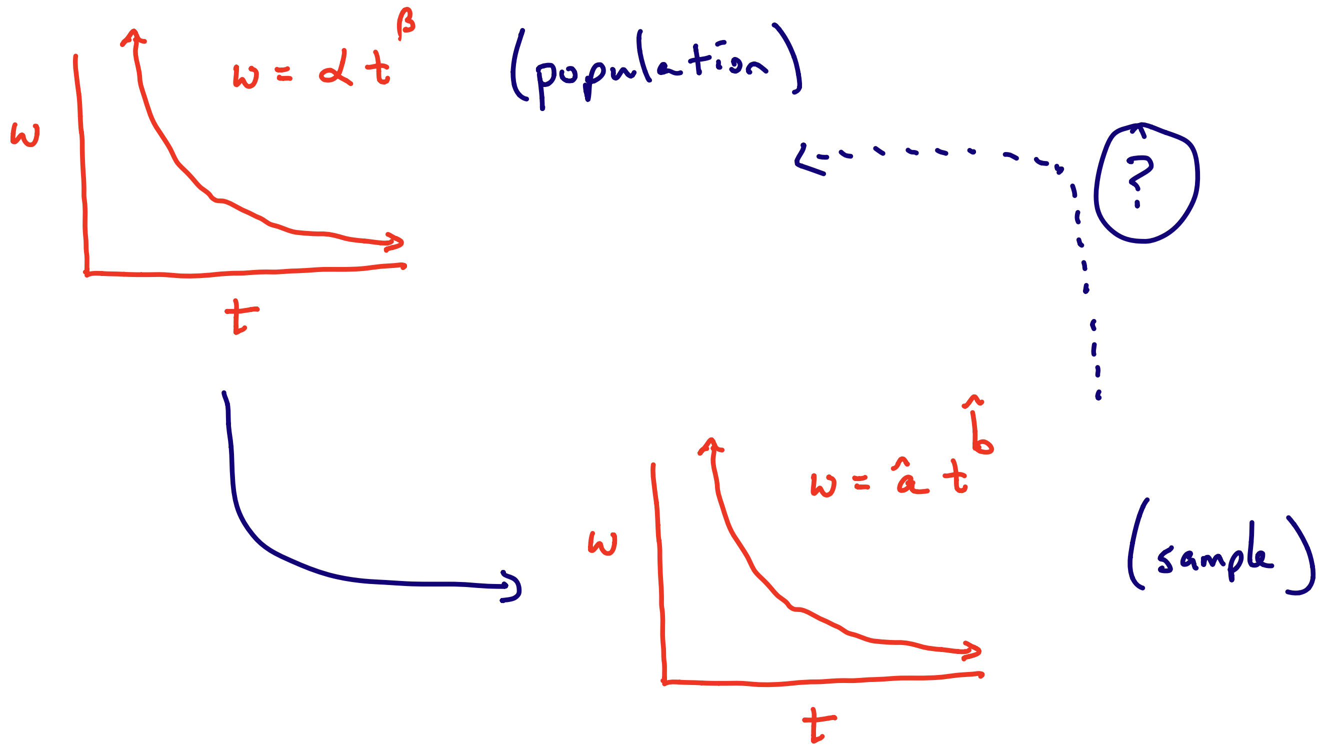
Lecture 5 – Parametric Bootstrapping

So far, we've learned how to estimate model parameters from observed data.

Problem of inference – how to infer population-level parameters from sample?



Our parameter estimates $\hat{a}$, $\hat{b}$ are just that: estimates.

How much variability can we expect in these estimates?

↳ need a "sampling distribution"

↳ use parametric bootstrapping

Parametric Bootstrapping

- concept: use Monte-Carlo simulation to construct the sampling distribution for α, β directly from our estimates $\hat{a}, \hat{b}$.

- Bootstrapping principle:

  * let $\hat{a}_i^b$ = i^{th} bootstrap estimate for α $(i = 1, 2, \cdots, N)$

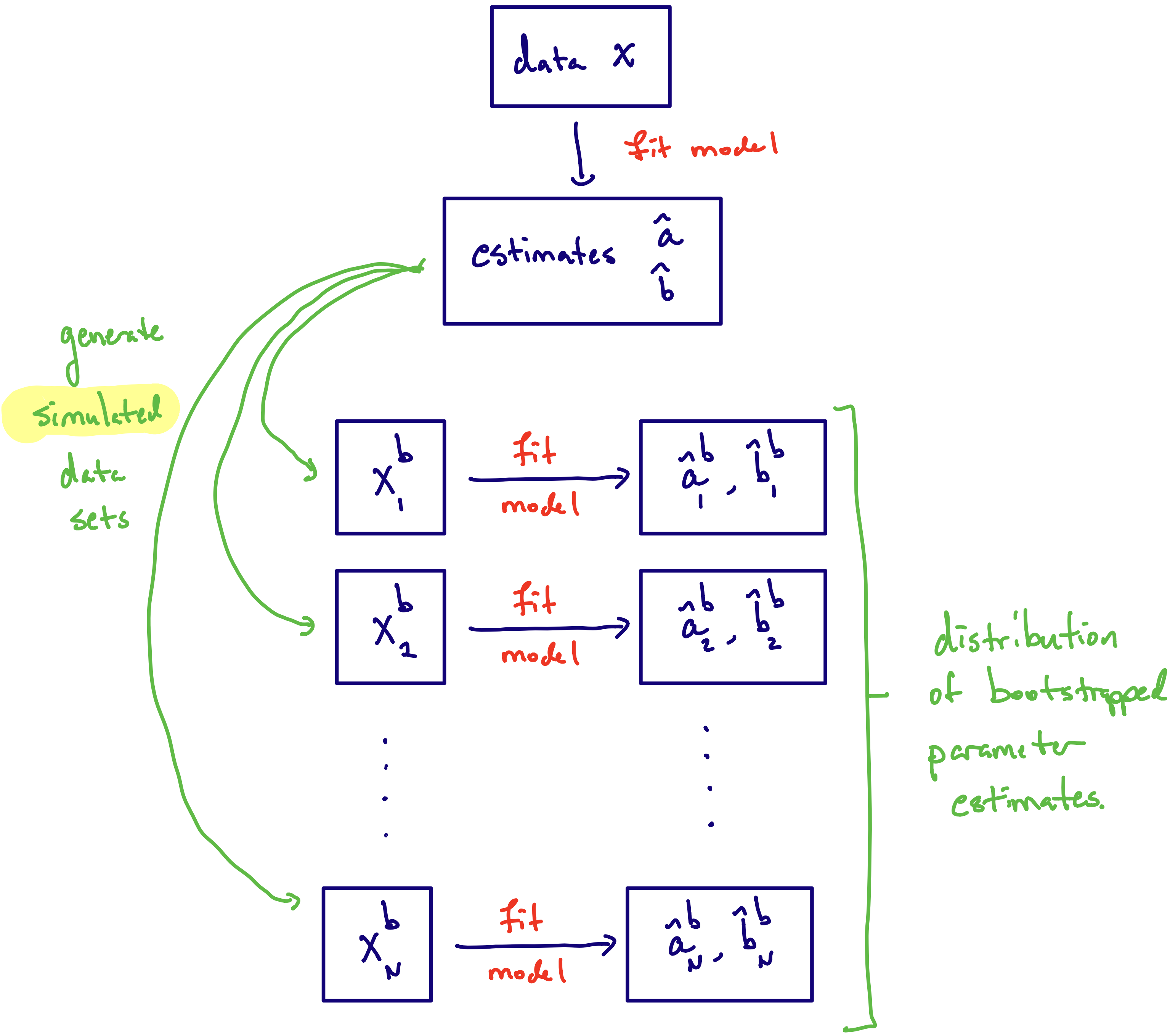
  * let A_N = distribution of bootstrap estimates $\hat{a}_i^b$

    A = "true" distribution of sample parameters a from generative model.

Then $A_N \longrightarrow A$ as $N \to \infty$.

↳ i.e, if we do a <u>lot</u> of bootstrap samples, our collection of bootstrap estimates will look very much like the true sampling distribution.

Schematic of method:



Rough outline of algorithm

1. get parameter estimates $\hat{a}, \hat{b}$
2. generate simulated data based on generative model with $\hat{a}, \hat{b}$ as parameters
3. fit model to simulated data – estimate $\hat{a}, \hat{b}$
4. do this many times!

Implementation in R

Let's assume we've already done MLE to fit model to observed data:

Preparation:

make this larger if adequate computing resources!

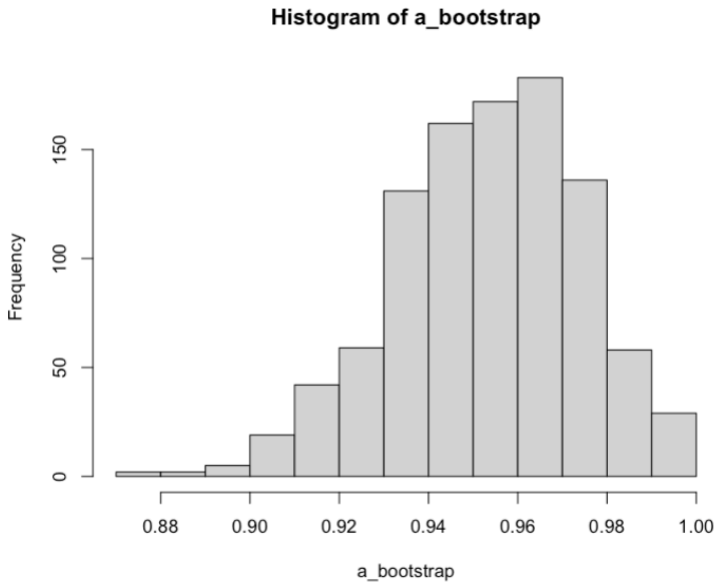
```r
# number of bootstrap samples
numSims = 1000

# extract initial parameter estimates
aHat = model$par[1]
bHat = model$par[2]

# set up empty vectors to store our bootstrapped estimates
a_bootstrap = numeric(numSims)
b_bootstrap = numeric(numSims)
```

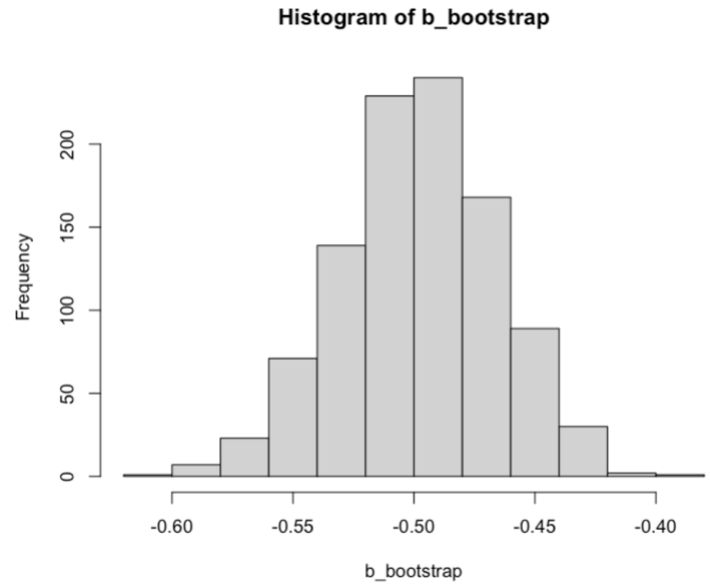
Perform bootstrapping:

```r
# do a loop
for (i in 1:numSims){
  # generate simulated data from binomial model from initial parameter estimates
  numRecall = rbinom(n=6, size=100, prob = aHat*times^bHat)
  X = data.frame(times, numRecall)

  # perform MLE in simulated dataset
  initPar = c(aHat, bHat) # best guess would be our MLE estimates!
  model = optim(par = initPar,
                fn = nll.power,
                data = X)

  # extract and store bootstrap parameter estimates
  a_bootstrap[i] = model$par[1]
  b_bootstrap[i] = model$par[2]
}
```

Analyze the distribution of bootstrap estimates:

```r
# look at bootstrapped parameter estimates
hist(a_bootstrap)
hist(b_bootstrap)
```





Use distribution of bootstrap estimates to construct 95% confidence intervals for α, β

```r
# 95% confidence intervals
quantile(a_bootstrap, probs = c(0.025, 0.975))
quantile(b_bootstrap, probs = c(0.025, 0.975))
```

these percentiles give "central" 95% CI