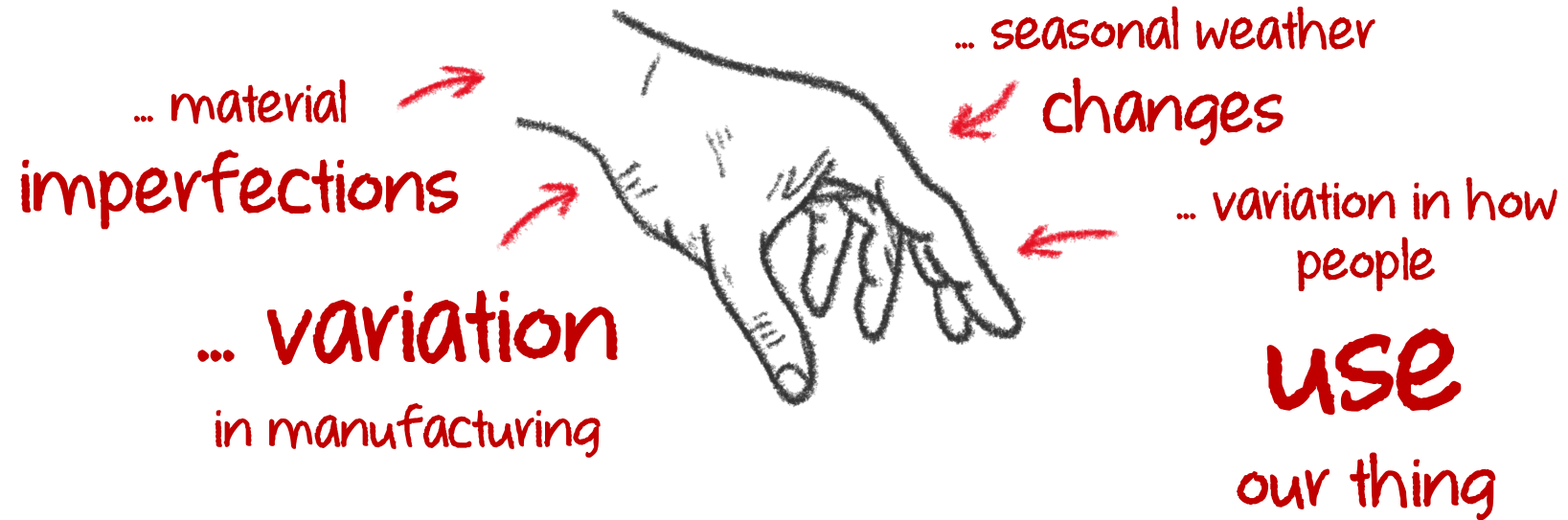


Reliability Analysis – Now What?

#3

... so what we did in
Reliability Analysis –
Now What #1 ...

the (random) hand of failure



time to failure (TTF)

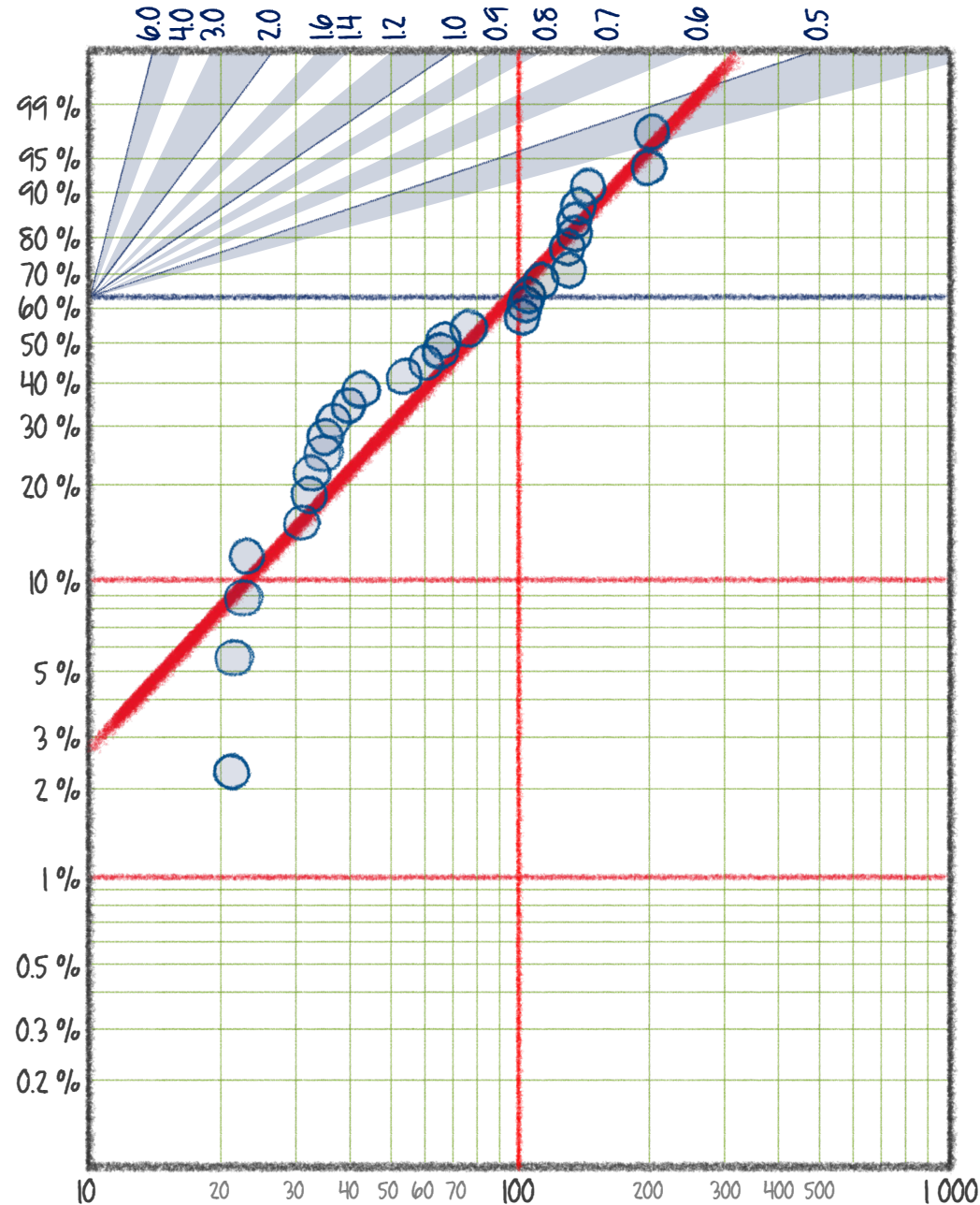
the (random) hand of failure



time to failure (TTF)

expected fraction
of your things that have failed

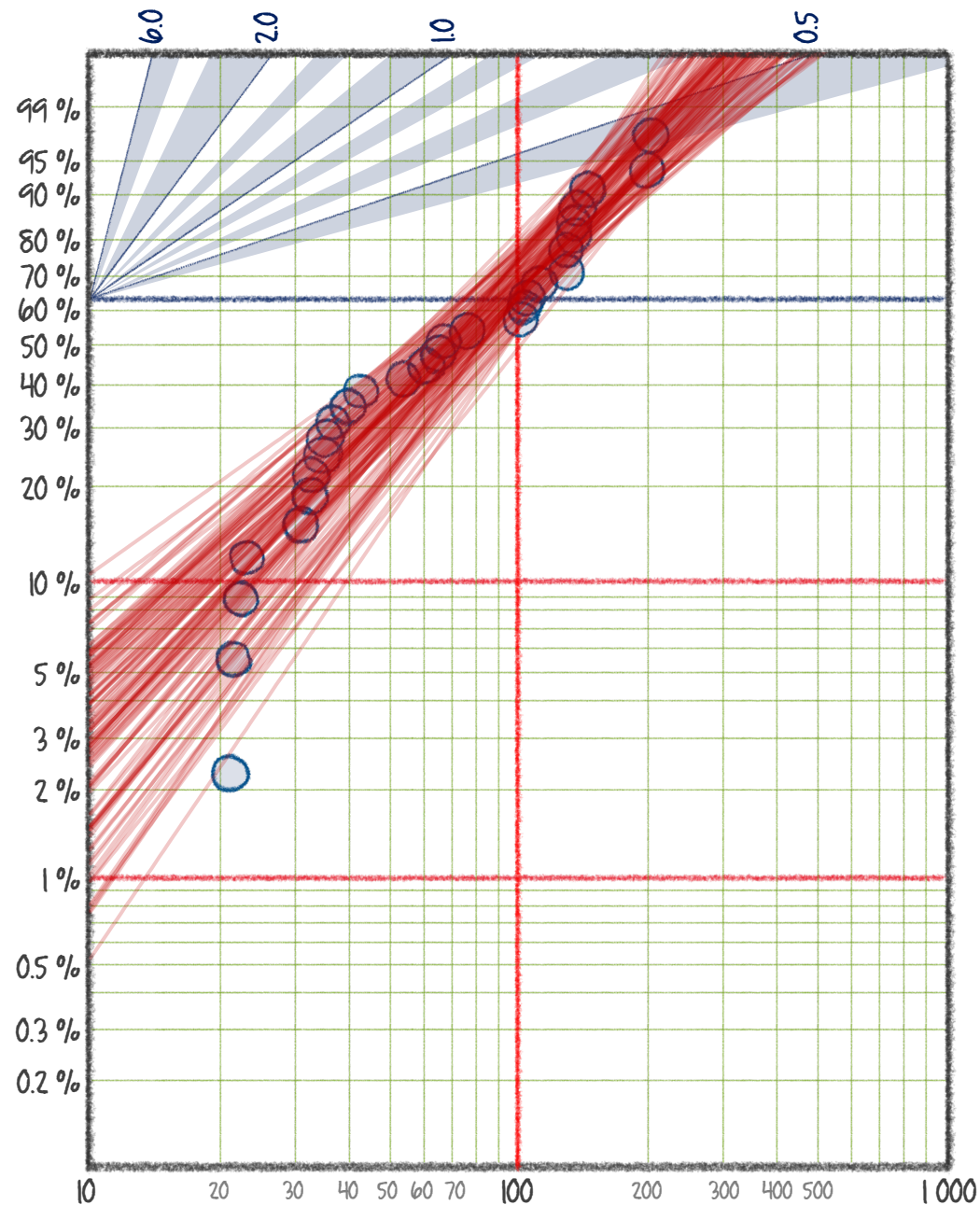
probability of failure



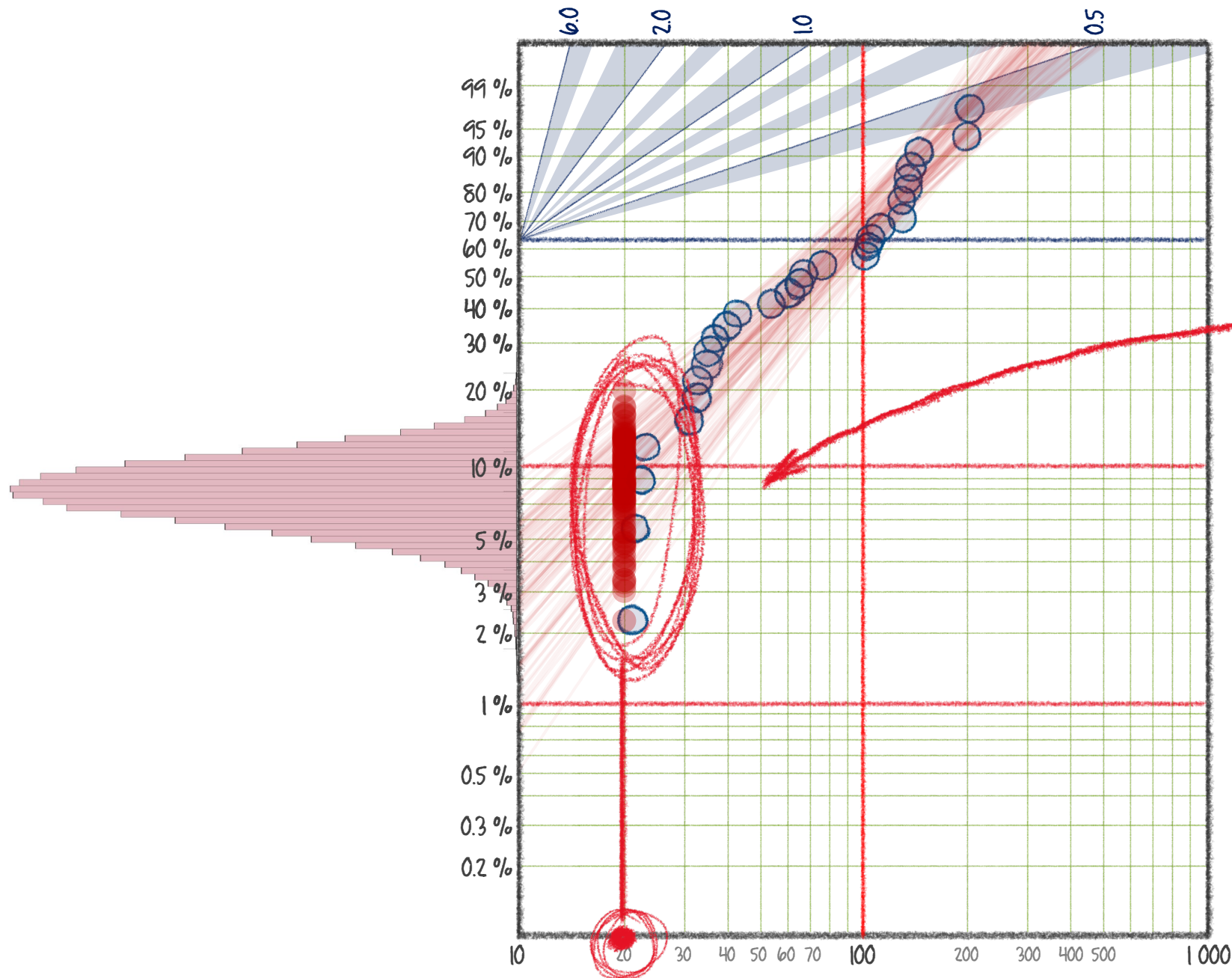
... these weird axes create

Weibull probability plotting paper

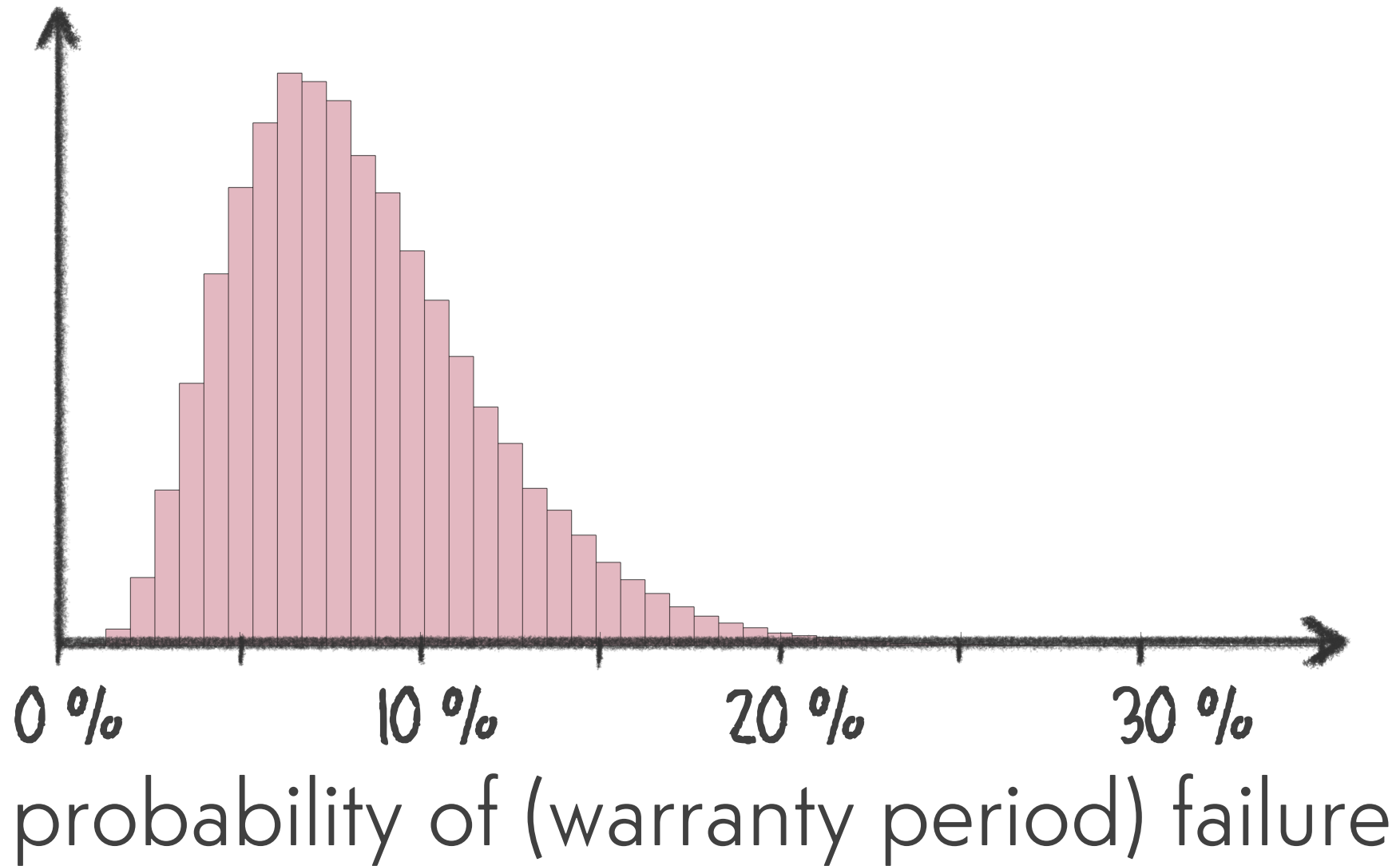
... and data will tend to a
straight line if it is
produced by a random
process described by a
Weibull distribution

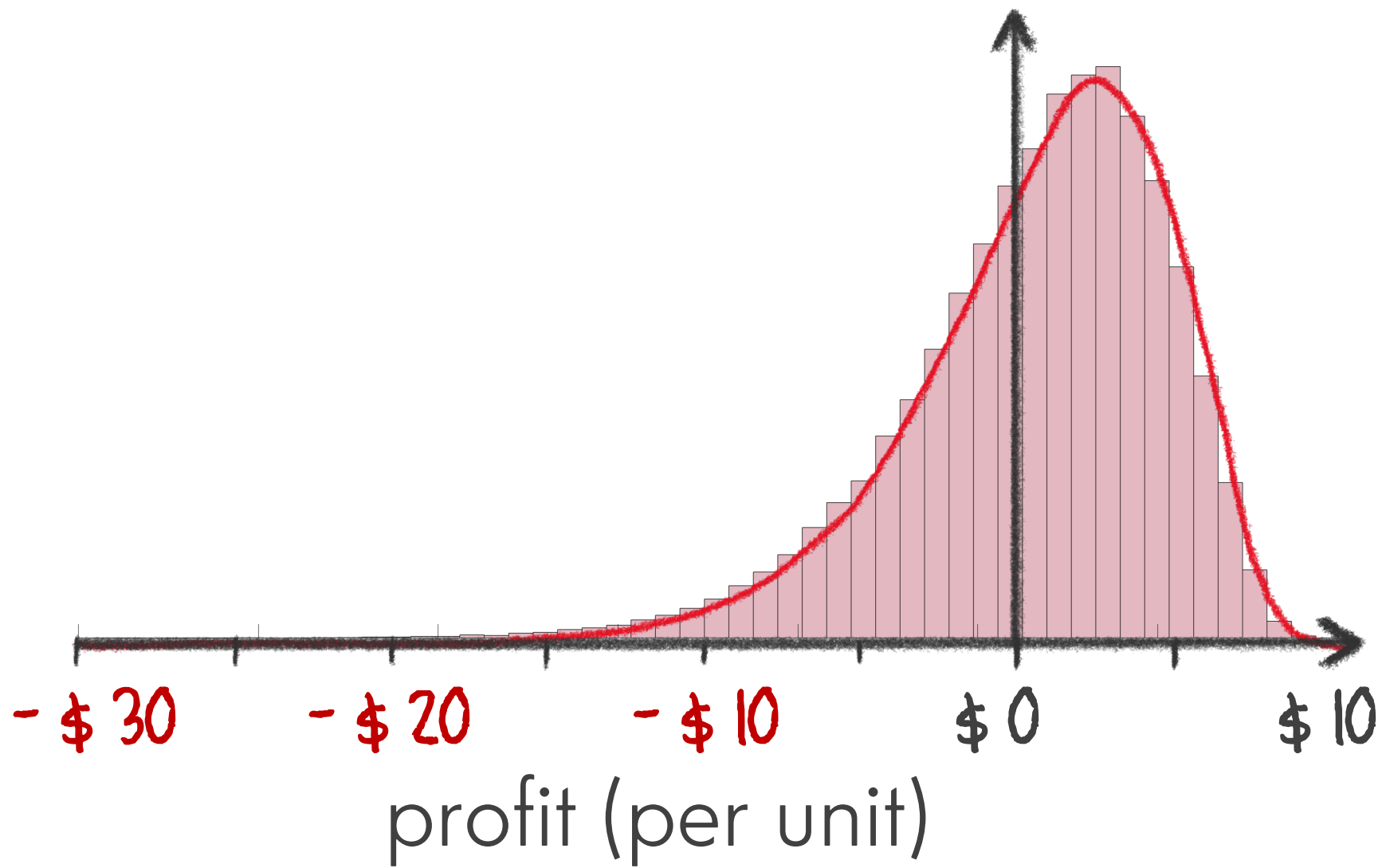


... we created a
posse
of Weibull
distributions



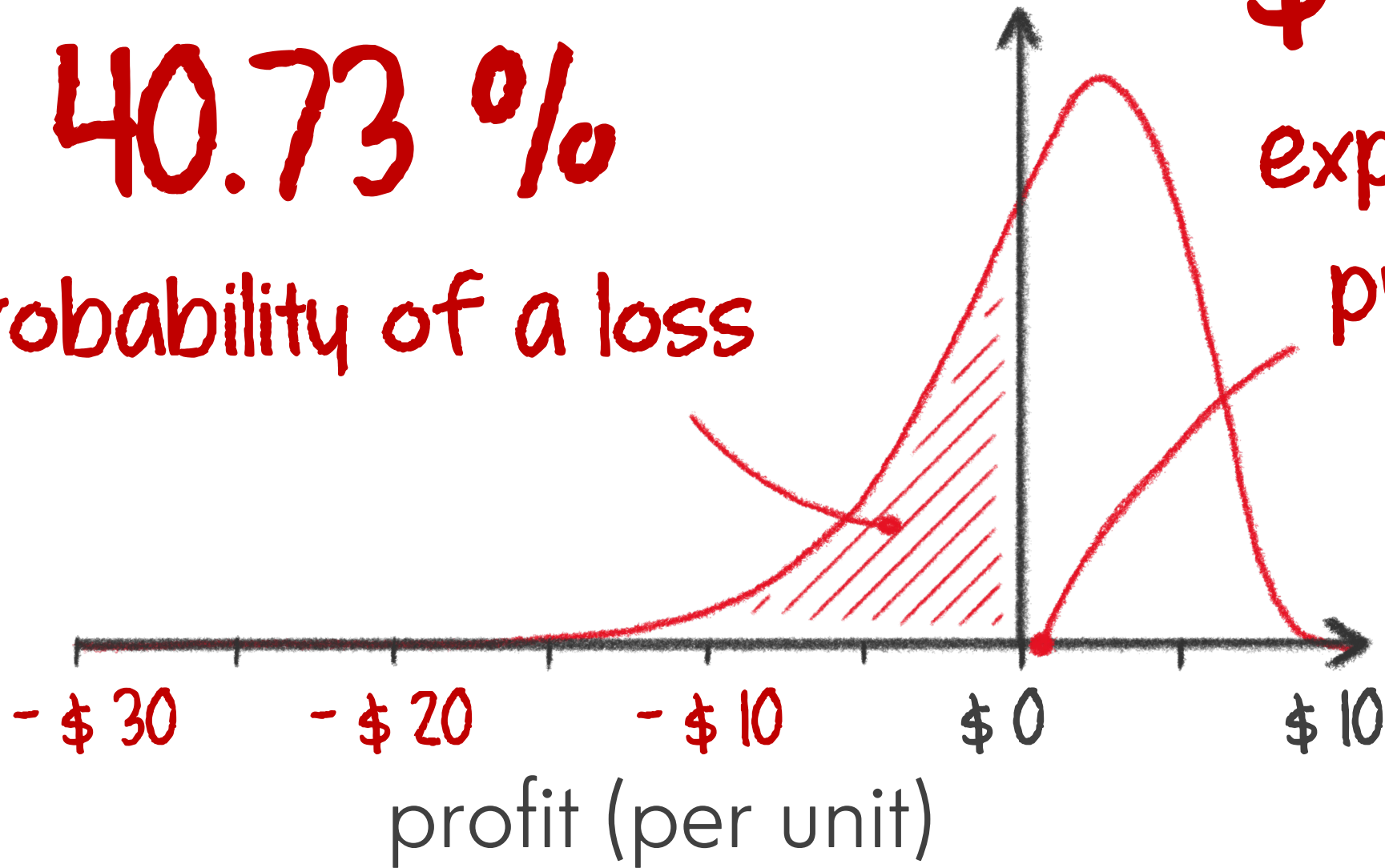
... which
means a
posse
of reliability
estimates at
2 years





40.73 %
probability of a loss

\$ 0.32
expected profit



... so what we did in
Reliability Analysis –
Now What #2 ...

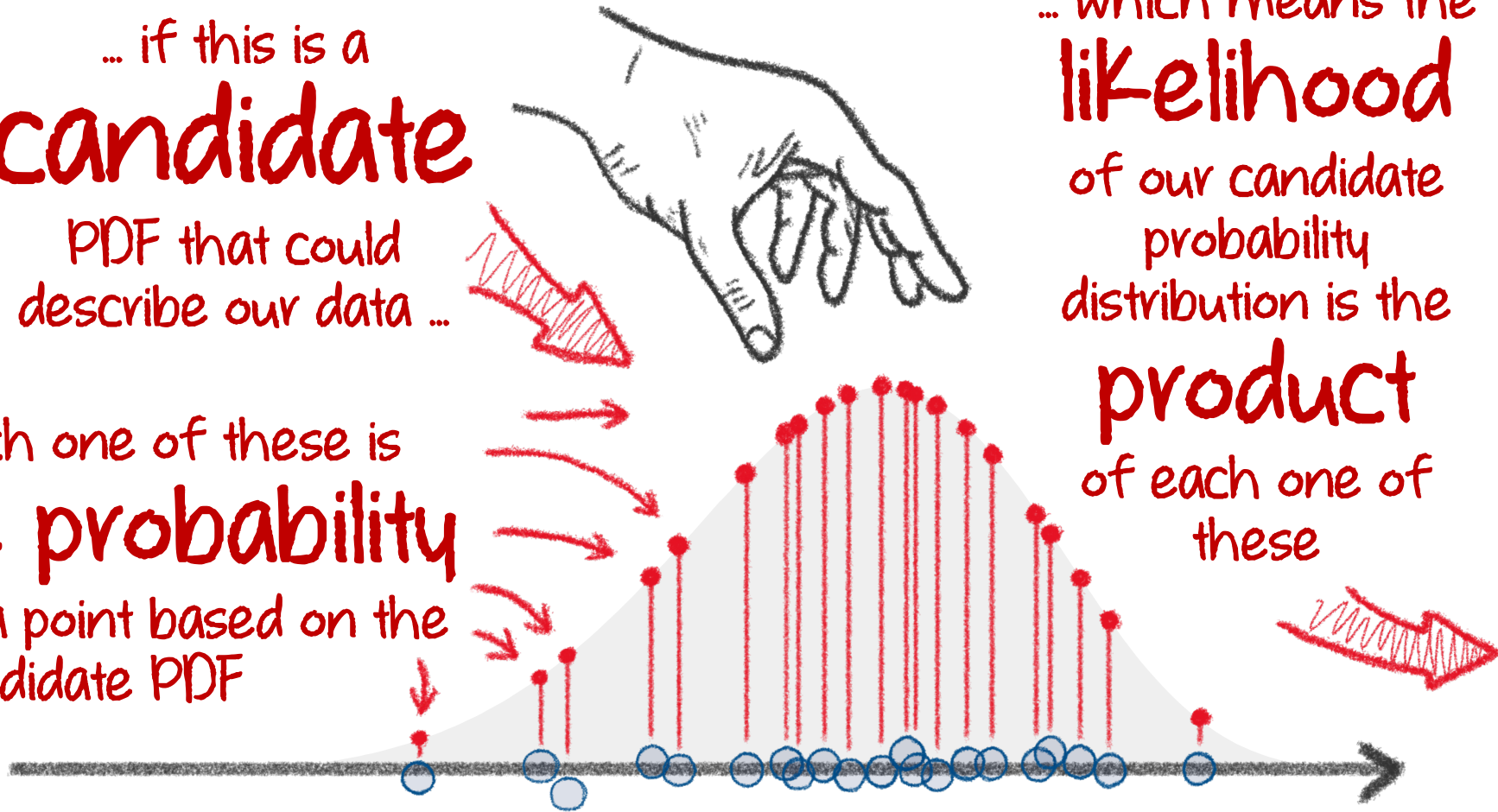
$f(x)$ or the
probability density function (PDF)
gives the relative probability
that a continuous random variable is
equal to a specific value

the (random) hand of failure

... if this is a
candidate

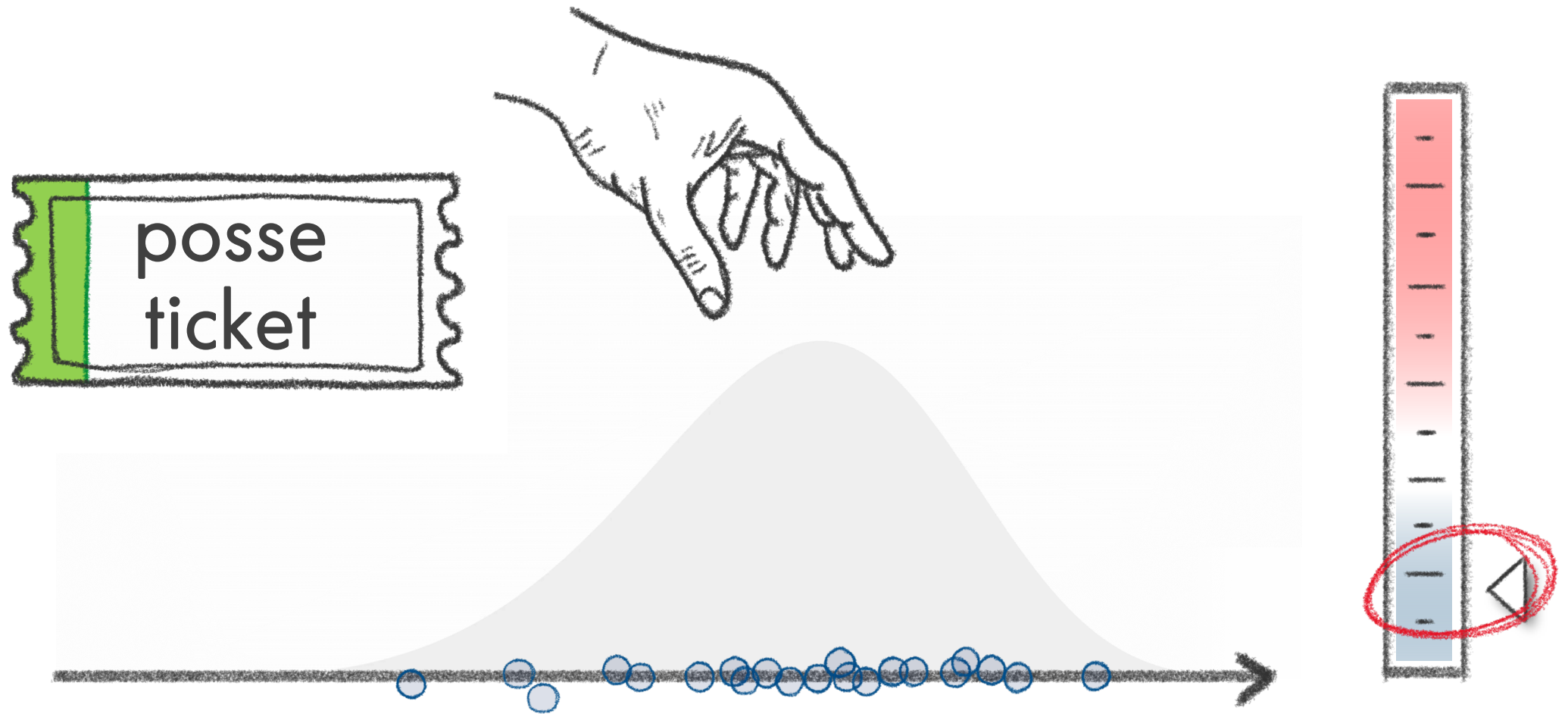
PDF that could
describe our data ...

... then each one of these is
relative probability
of each data point based on the
candidate PDF

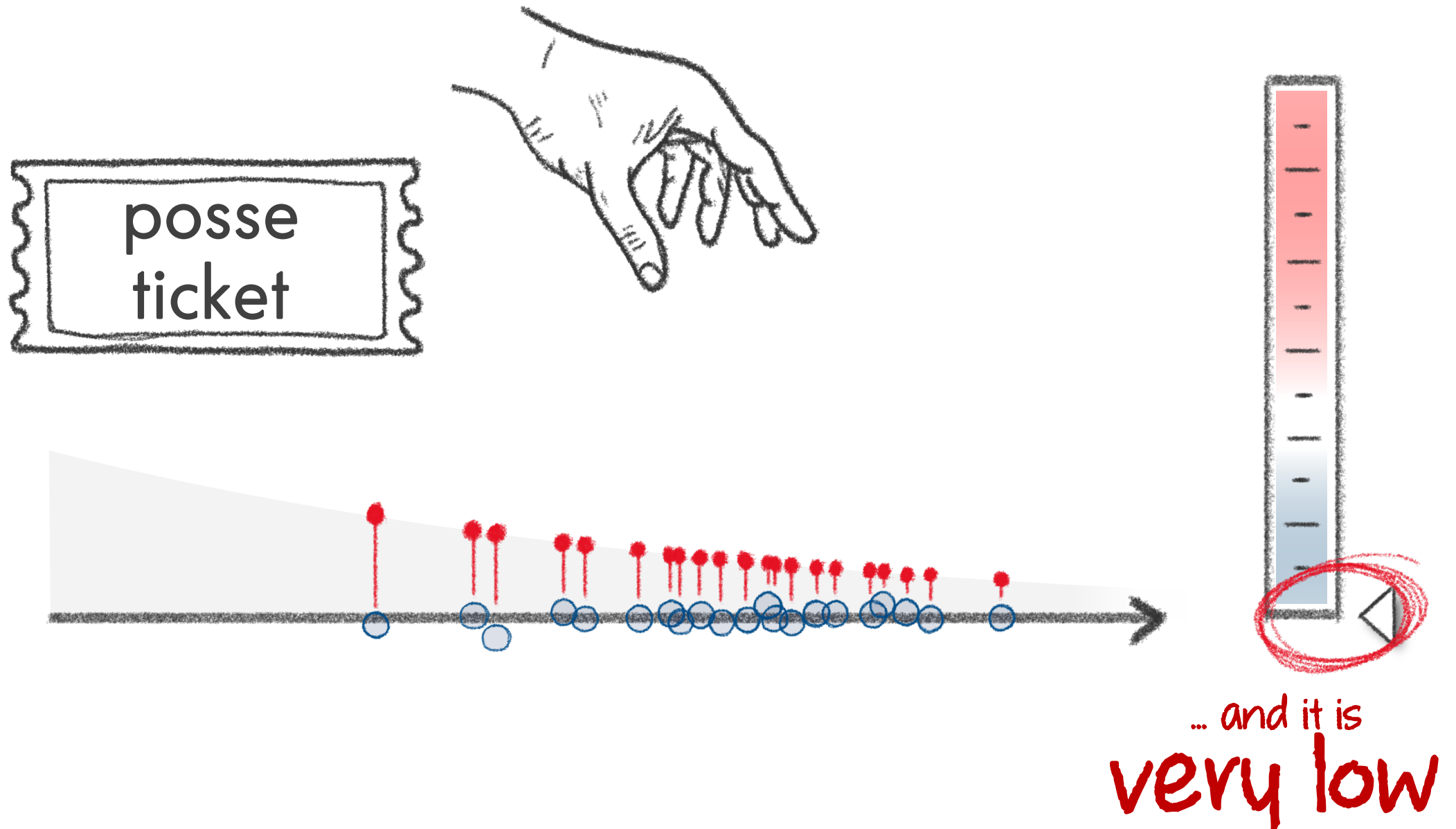


... which means the
likelihood
of our candidate
probability
distribution is the
product
of each one of
these

the (random) hand of failure



the (random) hand of failure



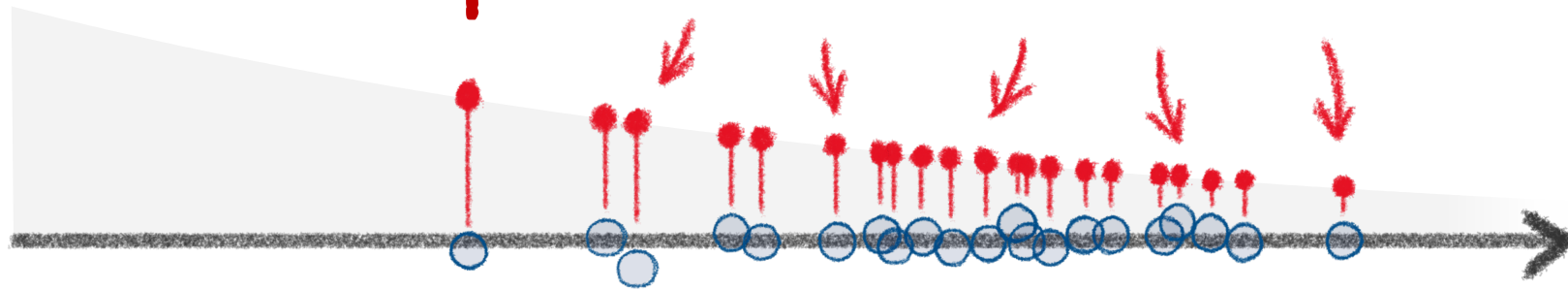
the (random) hand of failure

... the
likelihood

of this
candidate

PDF is the

product of these values



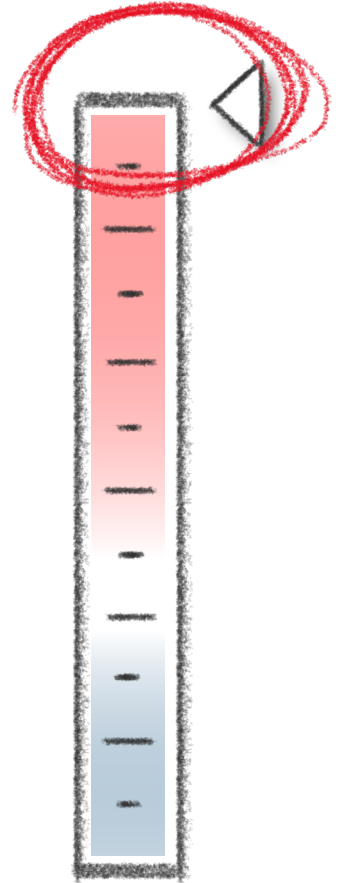
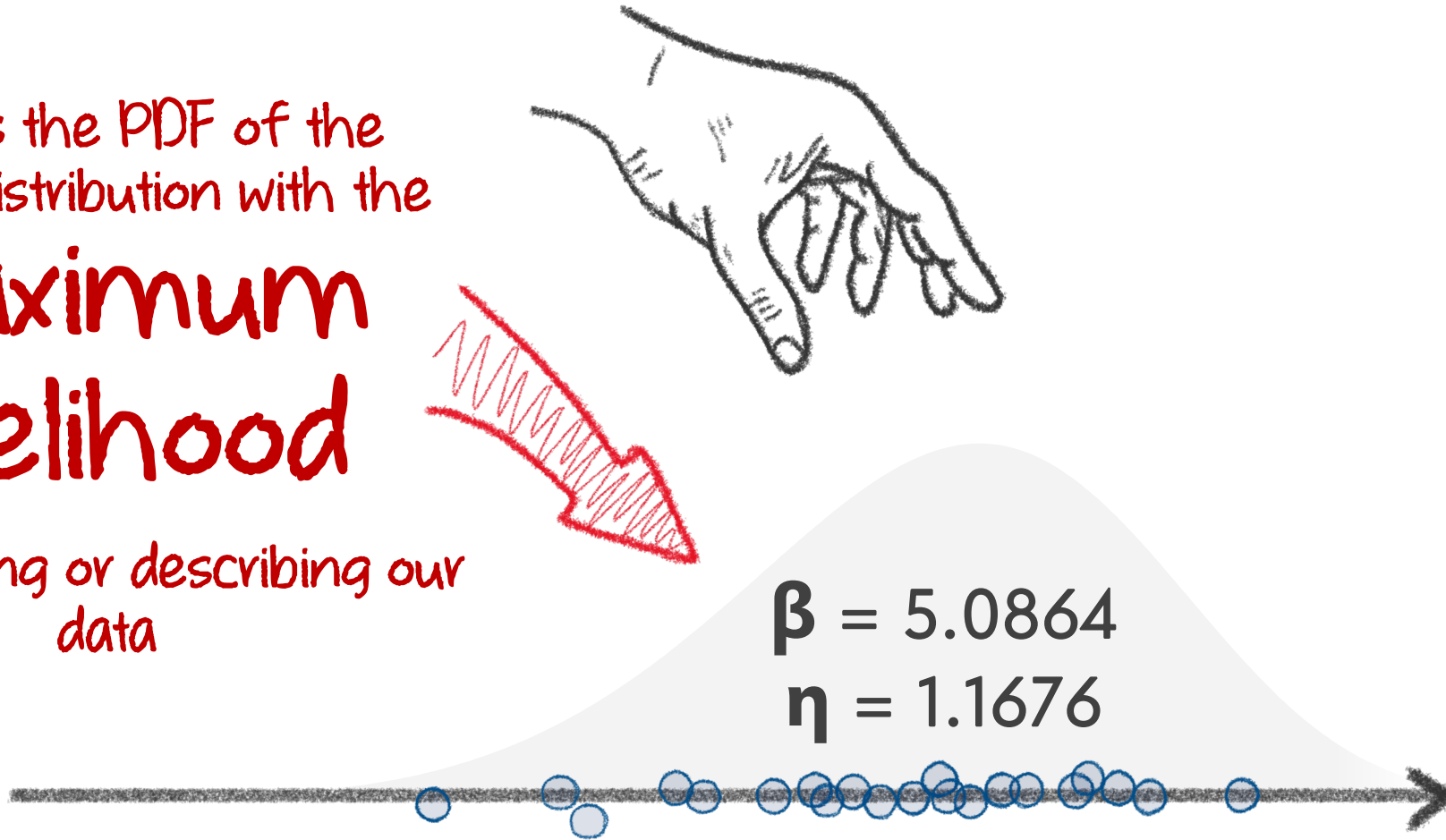
... and it is
very low

the (random) hand of failure

... this is the PDF of the
Weibull distribution with the

**maximum
likelihood**

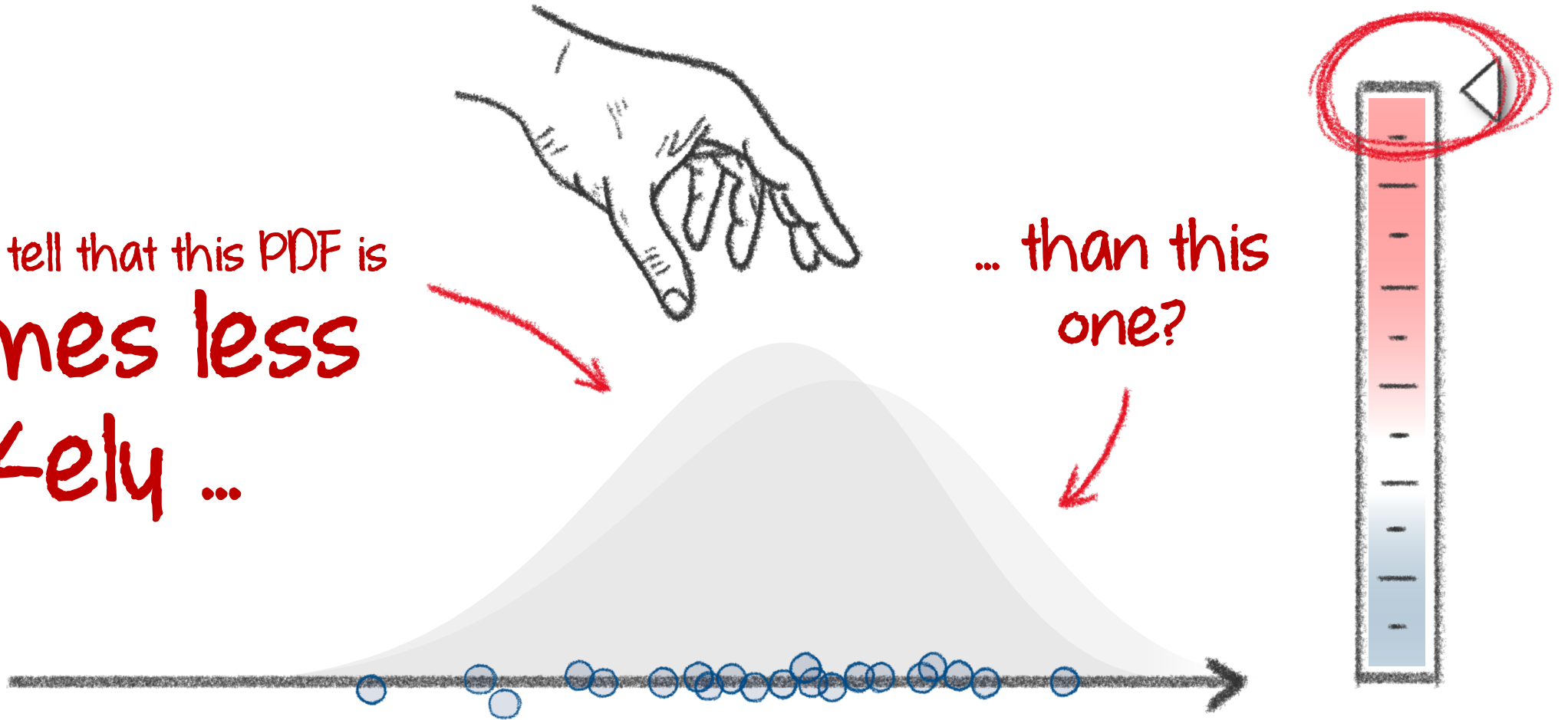
of explaining or describing our
data



the (random) hand of failure

... can you tell that this PDF is
**7 times less
likely ...**

... than this
one?



event	data	likelihood
fails at	t	$f(t)$
fails before	t	$F(t)$
fails after	t	$R(t)$
fails between	t^L and t^U	$F(t^U) - F(t^L)$

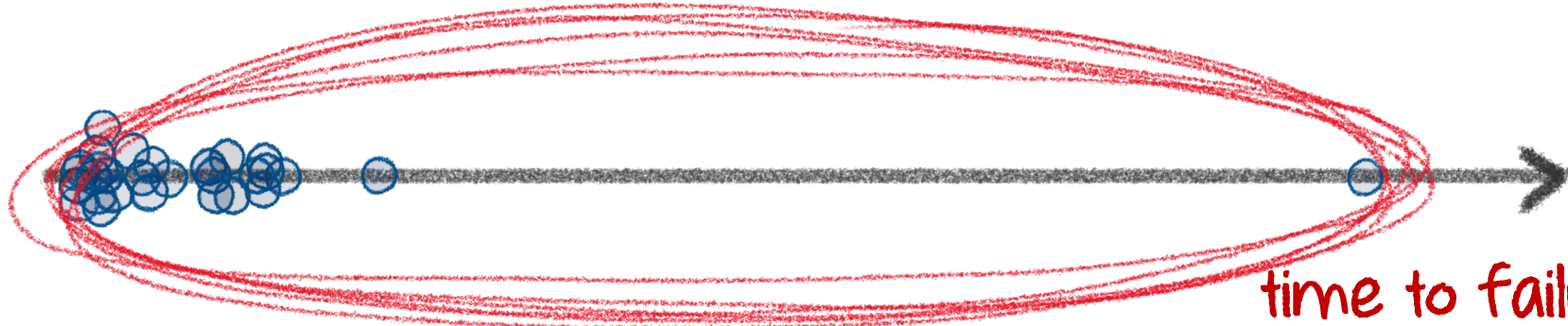
... so how do we get
our posse?

Markov Chain Monte Carlo Simulation (MCMC)

the (random) hand of failure

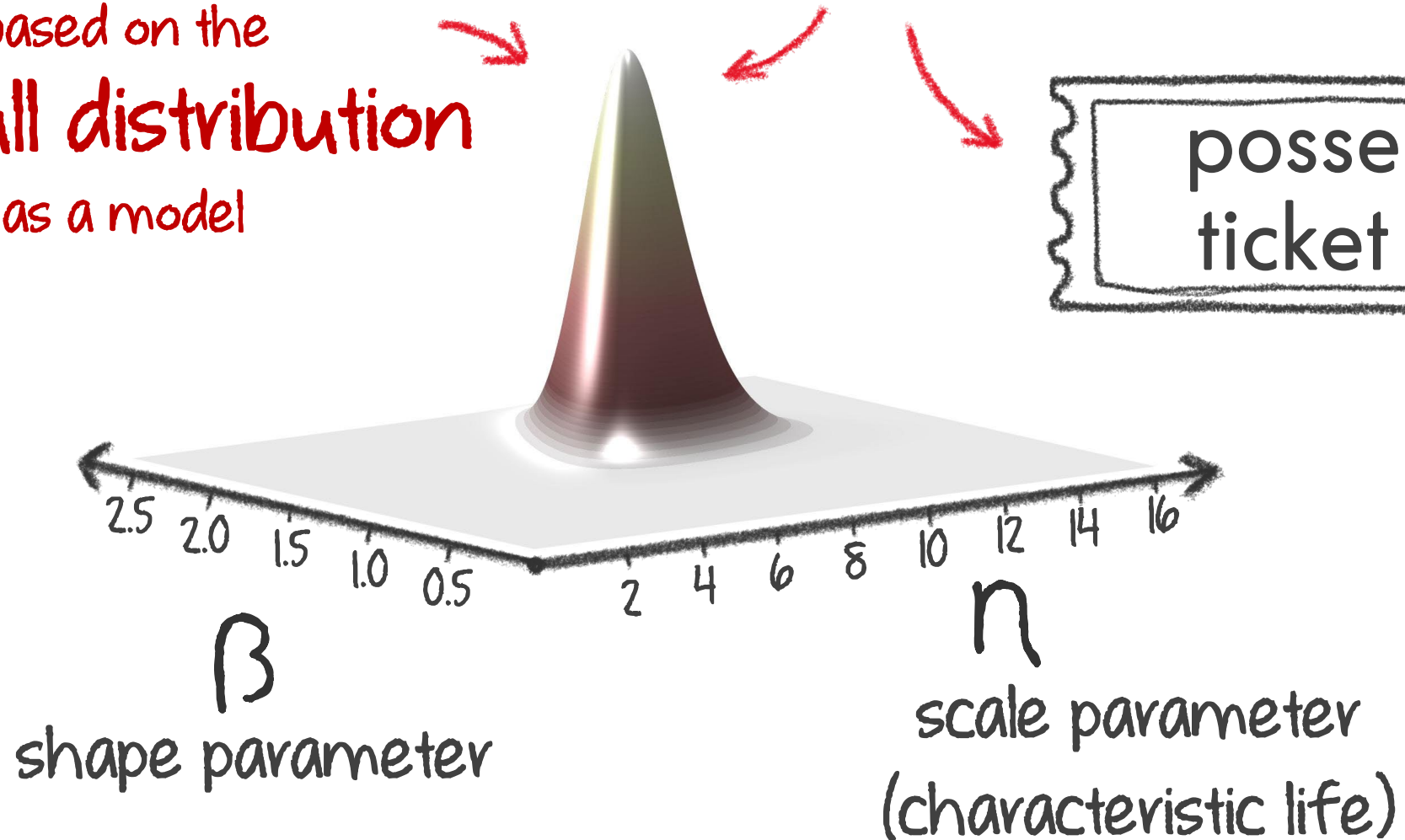
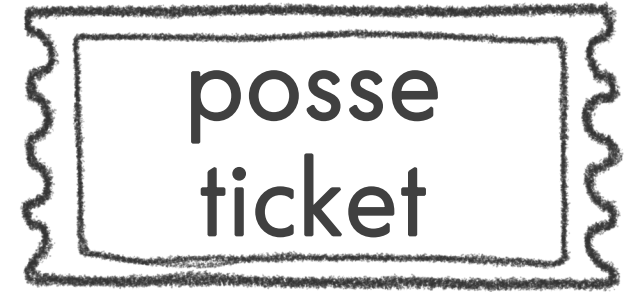


DATA



this is the
likelihood function
based on the
Weibull distribution
as a model

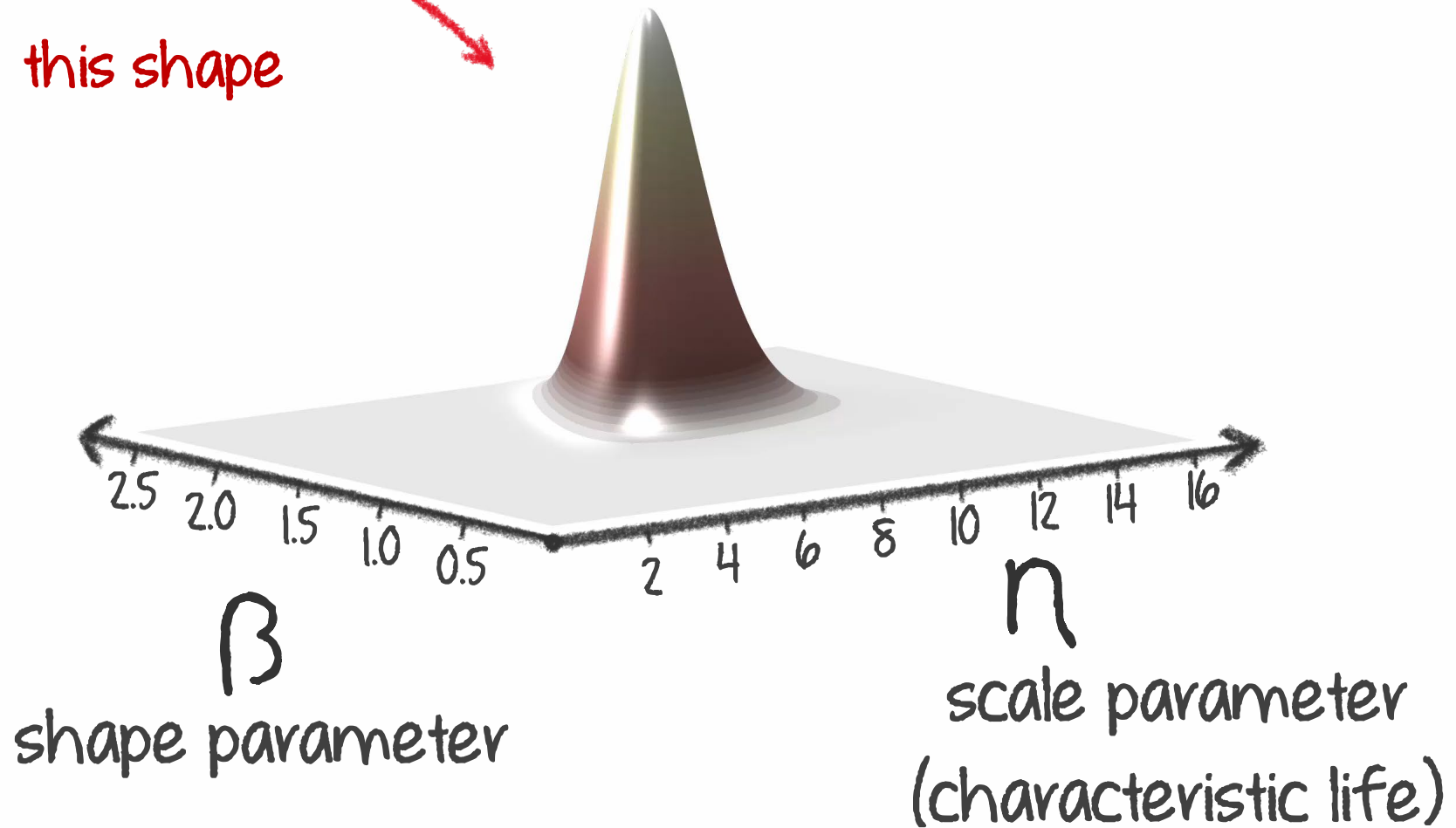
$L(\text{all data})$

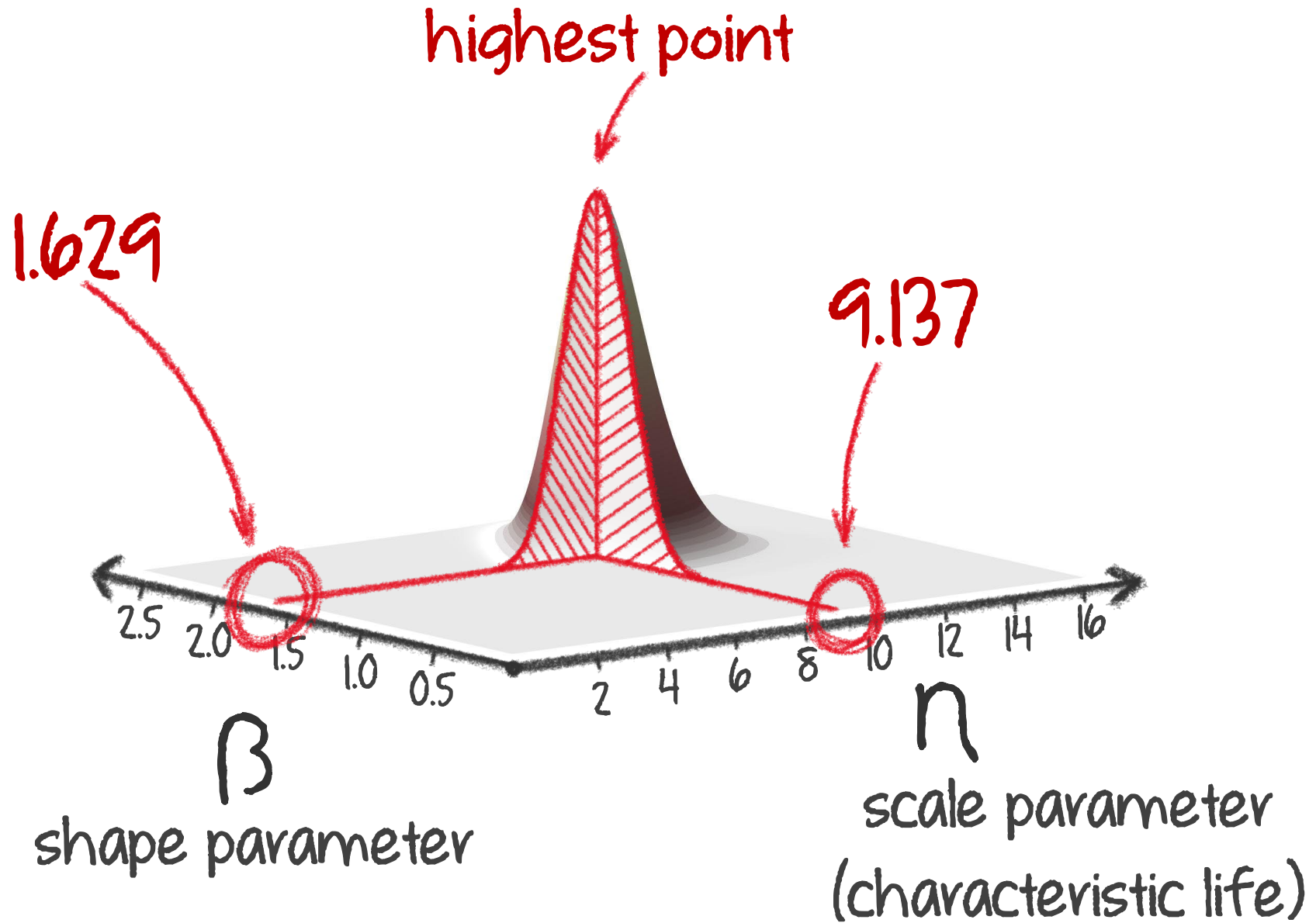


computers can't

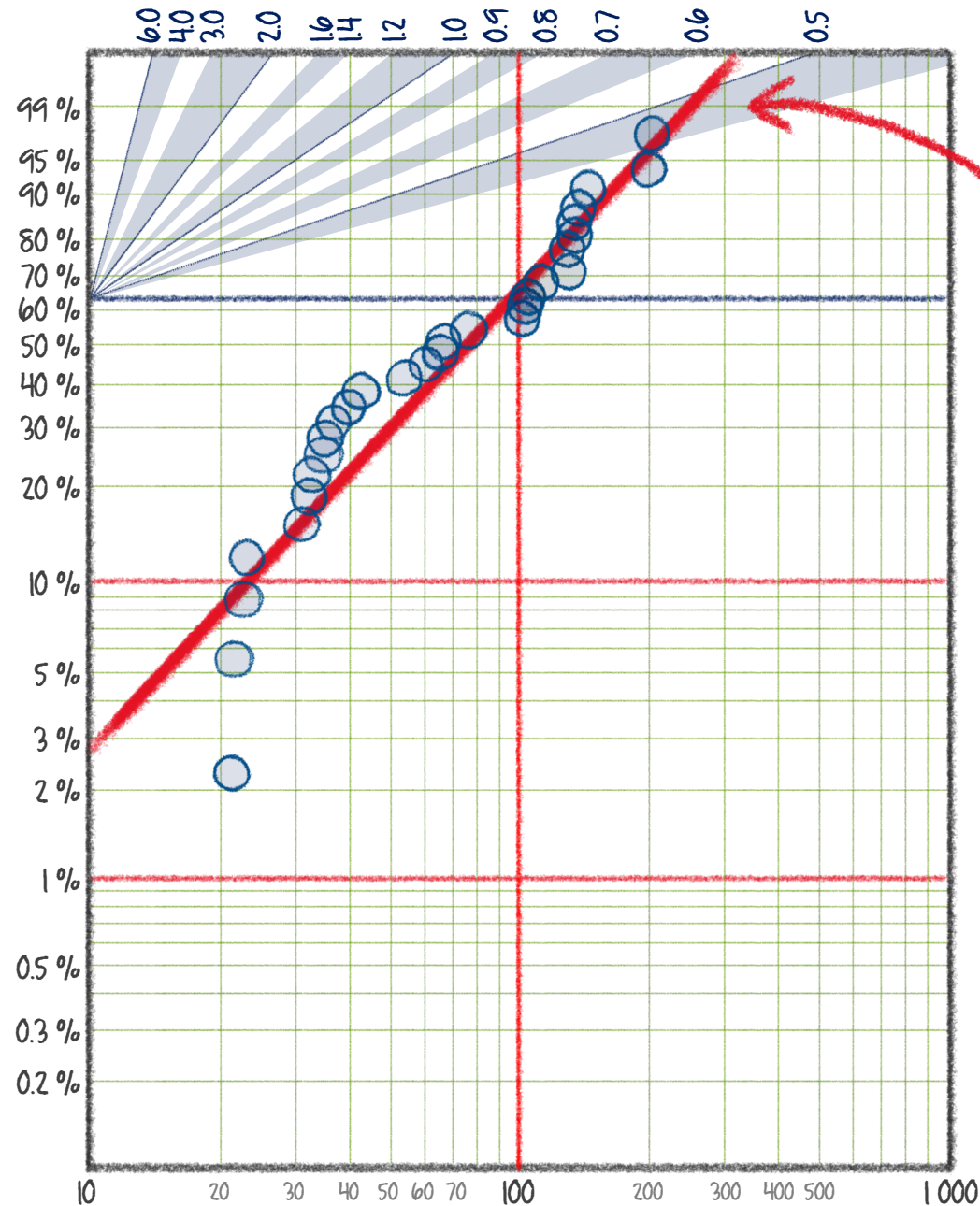
see

this shape





probability of failure



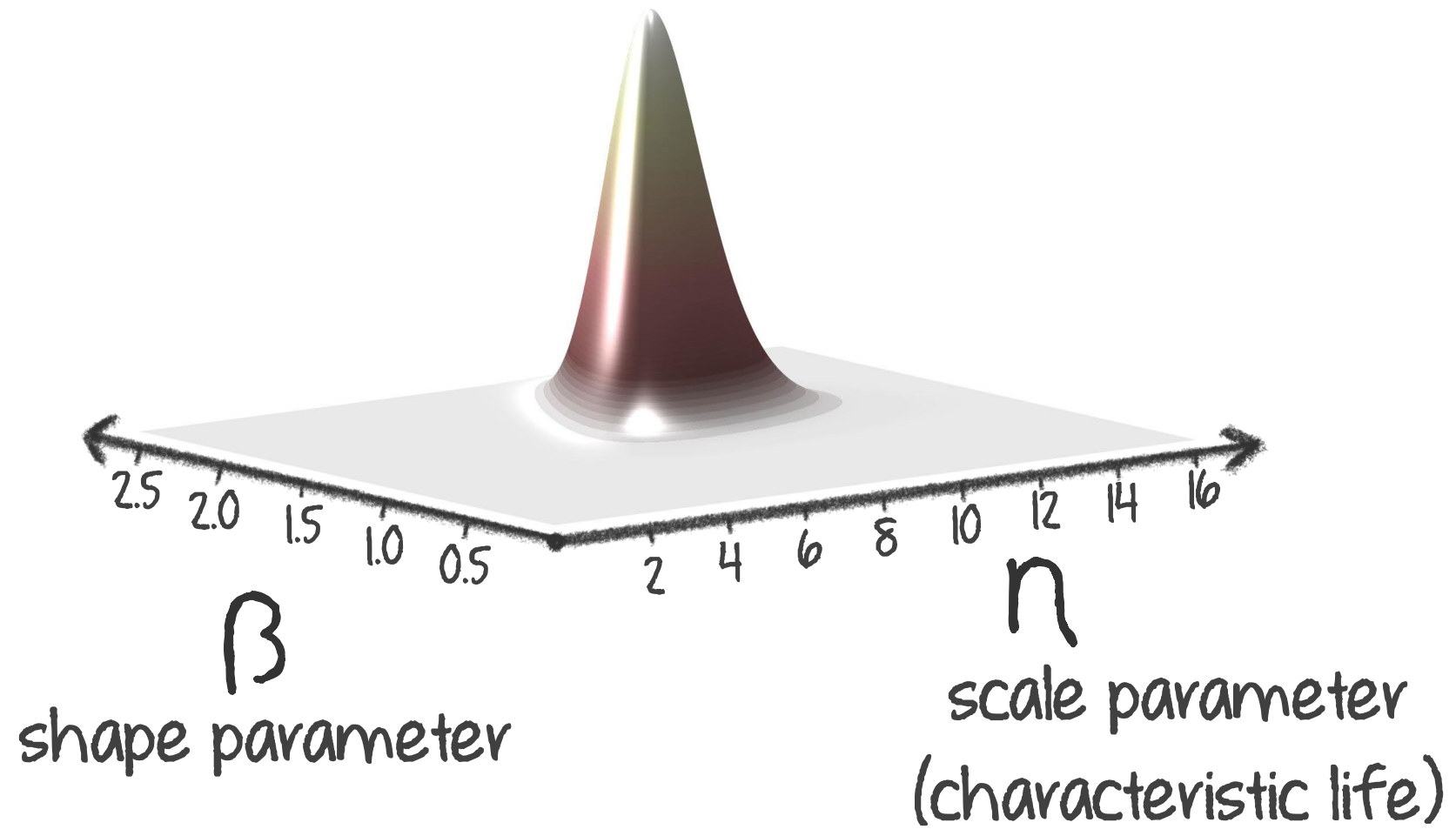
... Weibull distribution
CDF where ...

$$\beta = 1.629$$

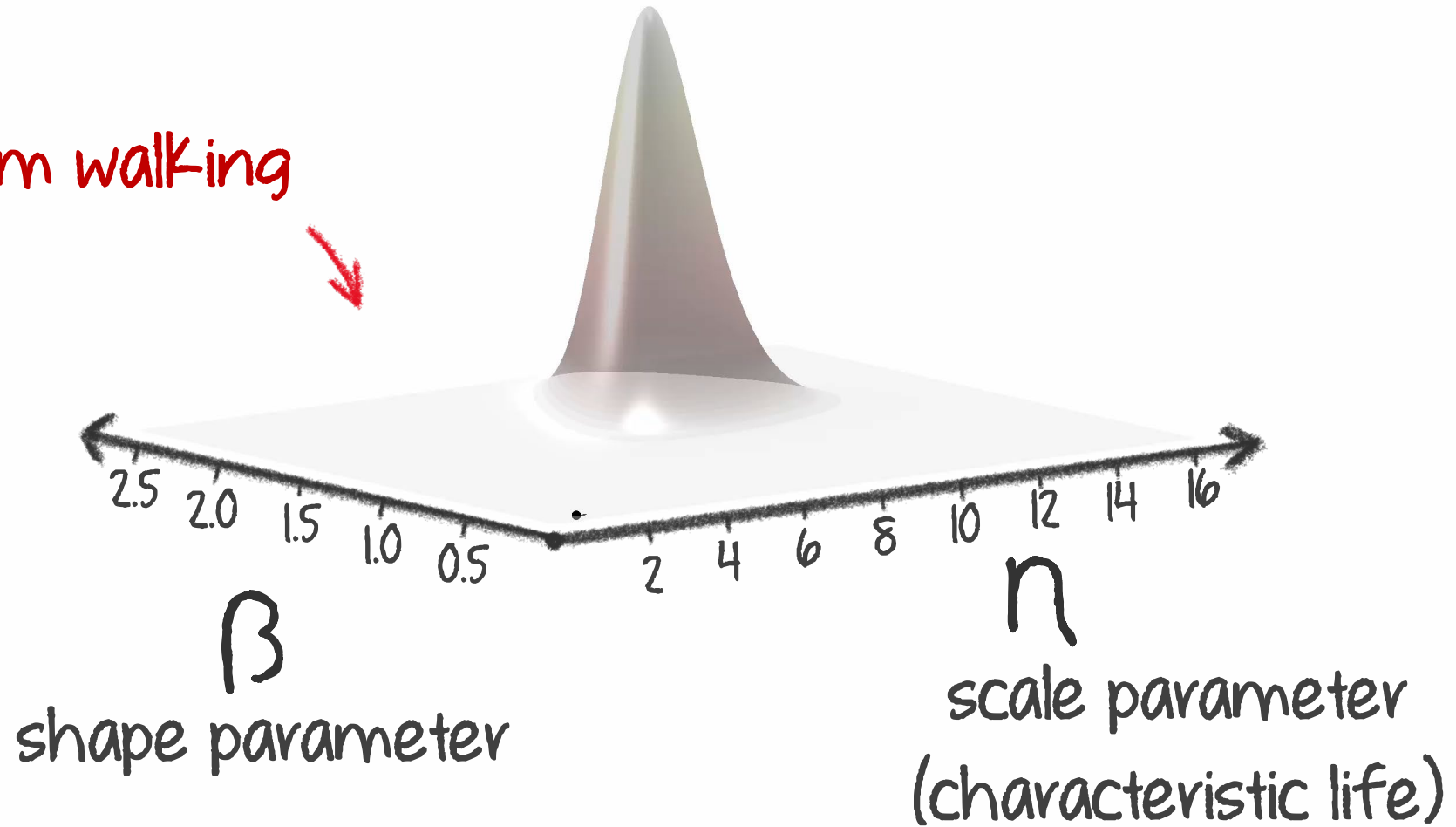
$$\eta = 9.137$$

... and our posse?

one approach is the
'classic' random walk
algorithm

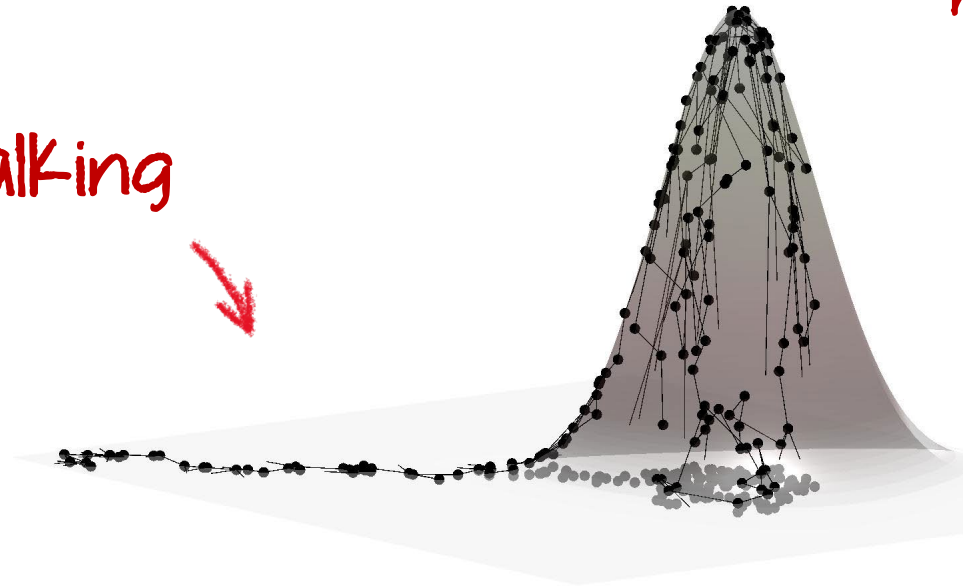


... random walking



... we take a random step
in a random direction ...

... random walking



... then (randomly) pick a
number between 0 and 1

... and if the likelihood of
your new point ...

... divided by the likelihood
of your old point ...

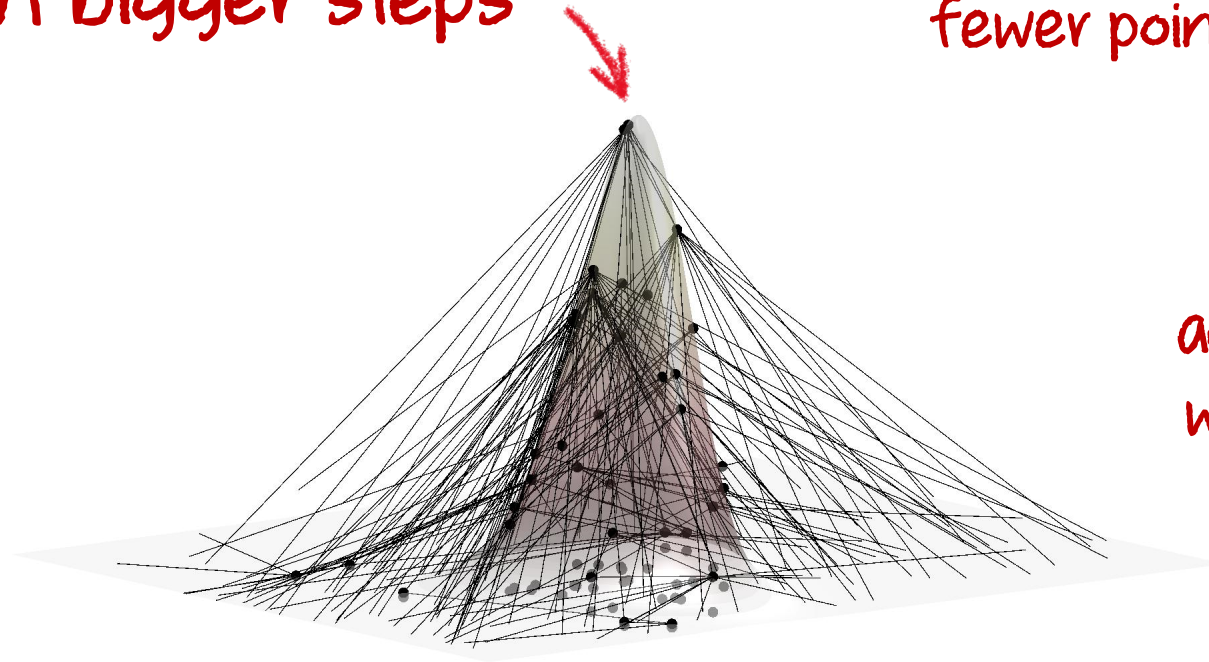
... is greater than ...

... we finish that step

... otherwise we stay at
the old point

... random walking
with bigger steps

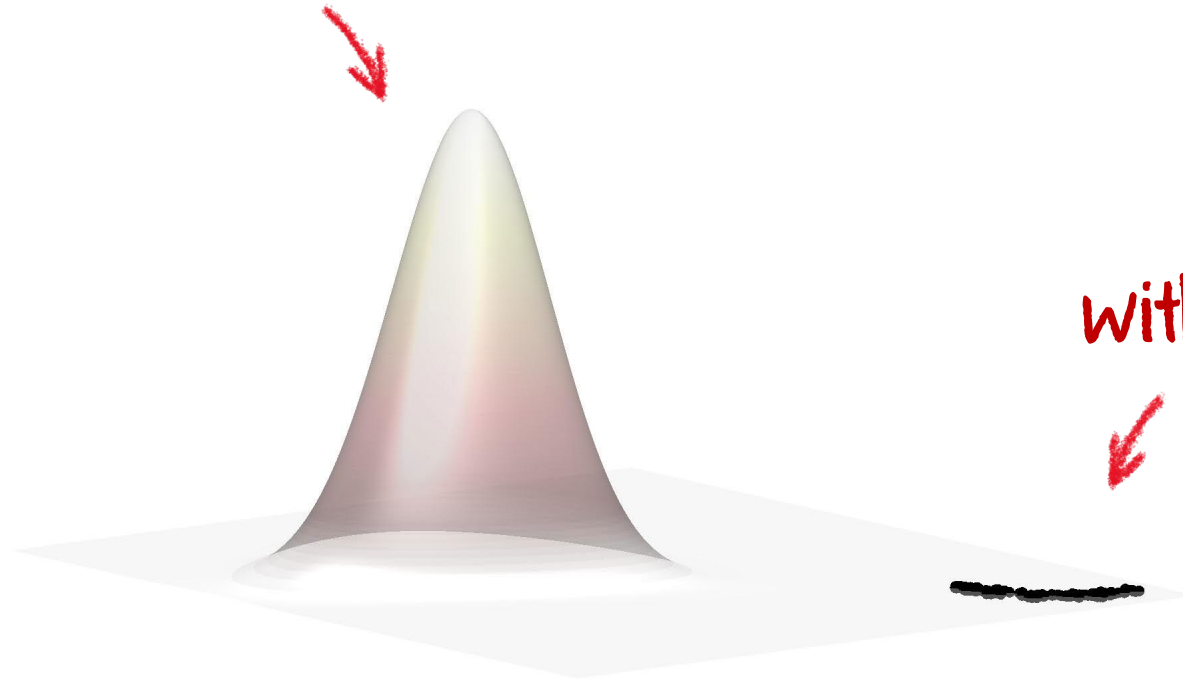
... it looks like
fewer points



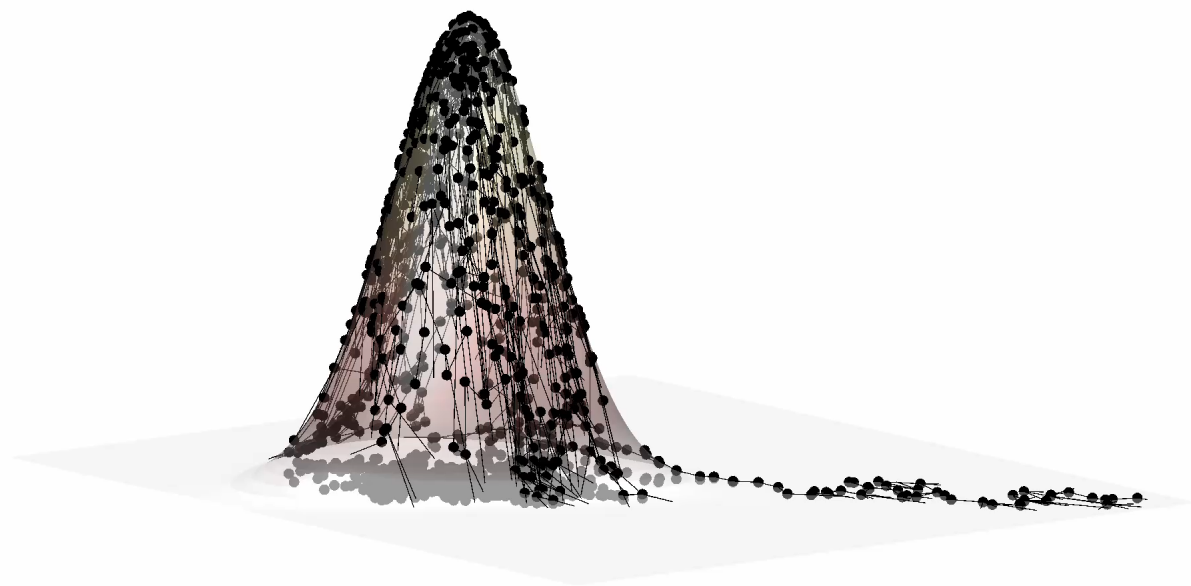
... but there are
actually lots of points
with the same value

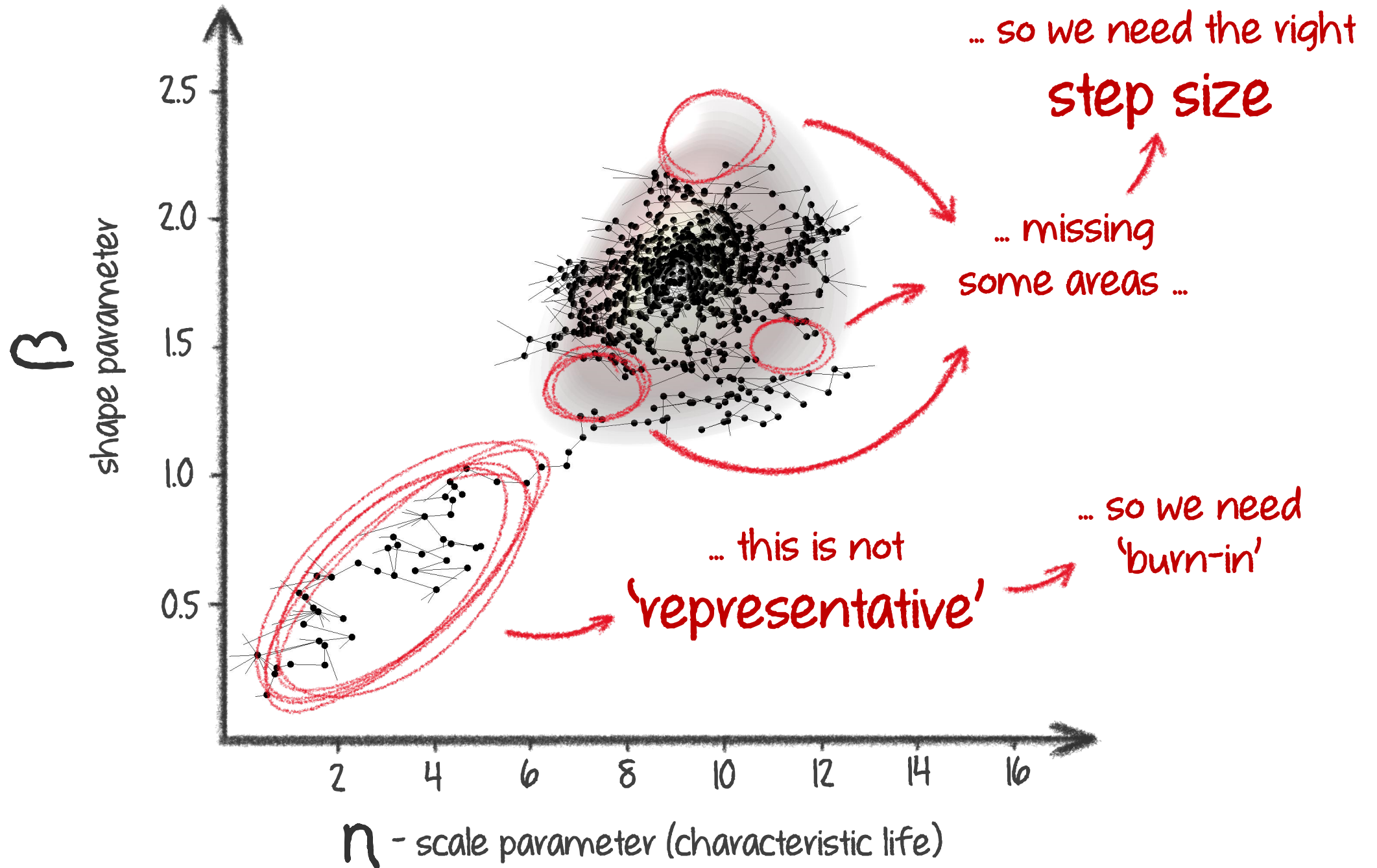
... which is not the best ...

... so we never explore all the
likelihood function



... random walking
with smaller steps





... another way?

but before I do ...

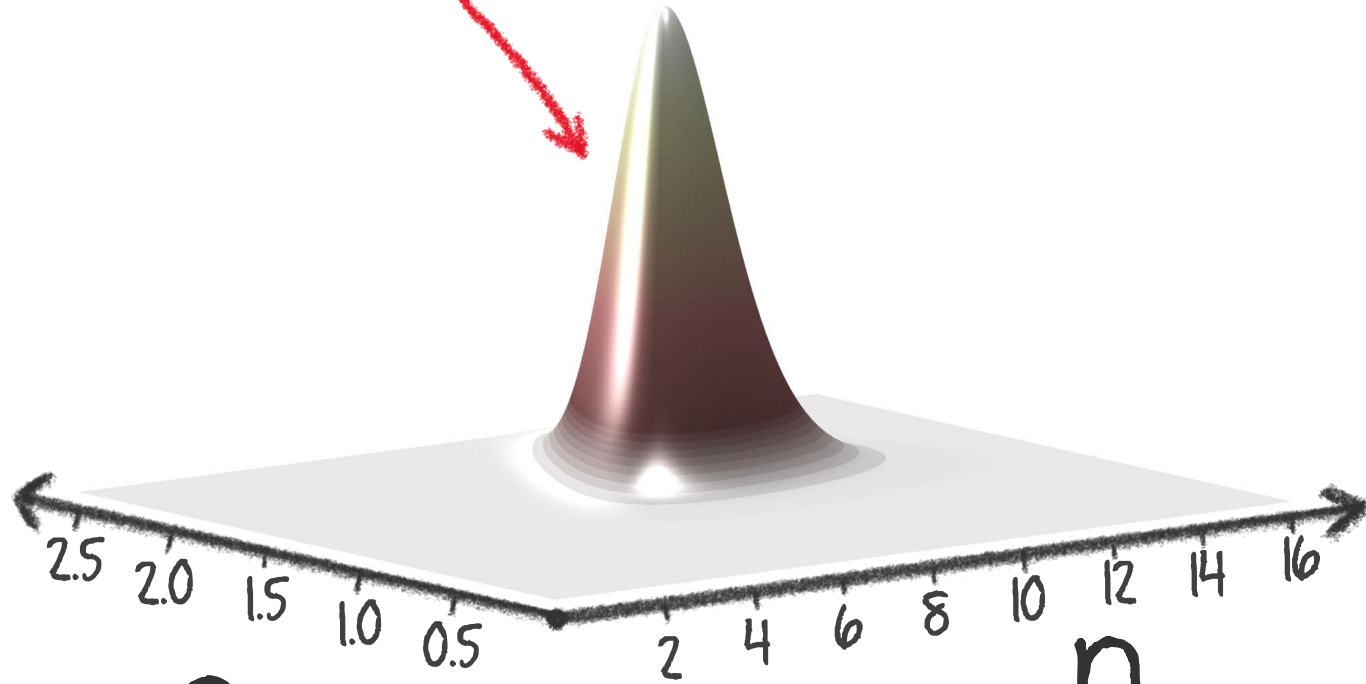
... I need to introduce ...

the rectangle of guess

computers can't

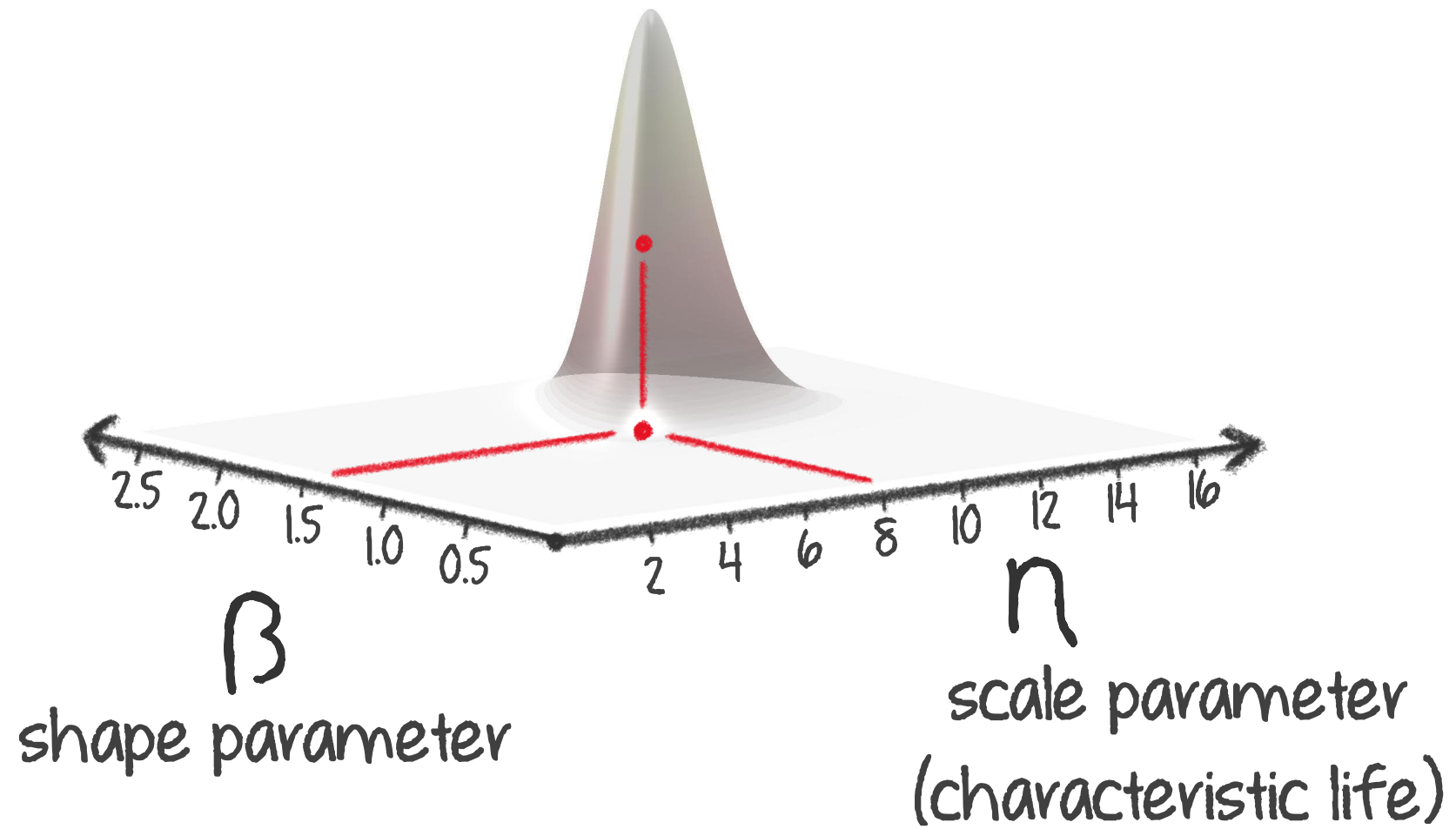
see

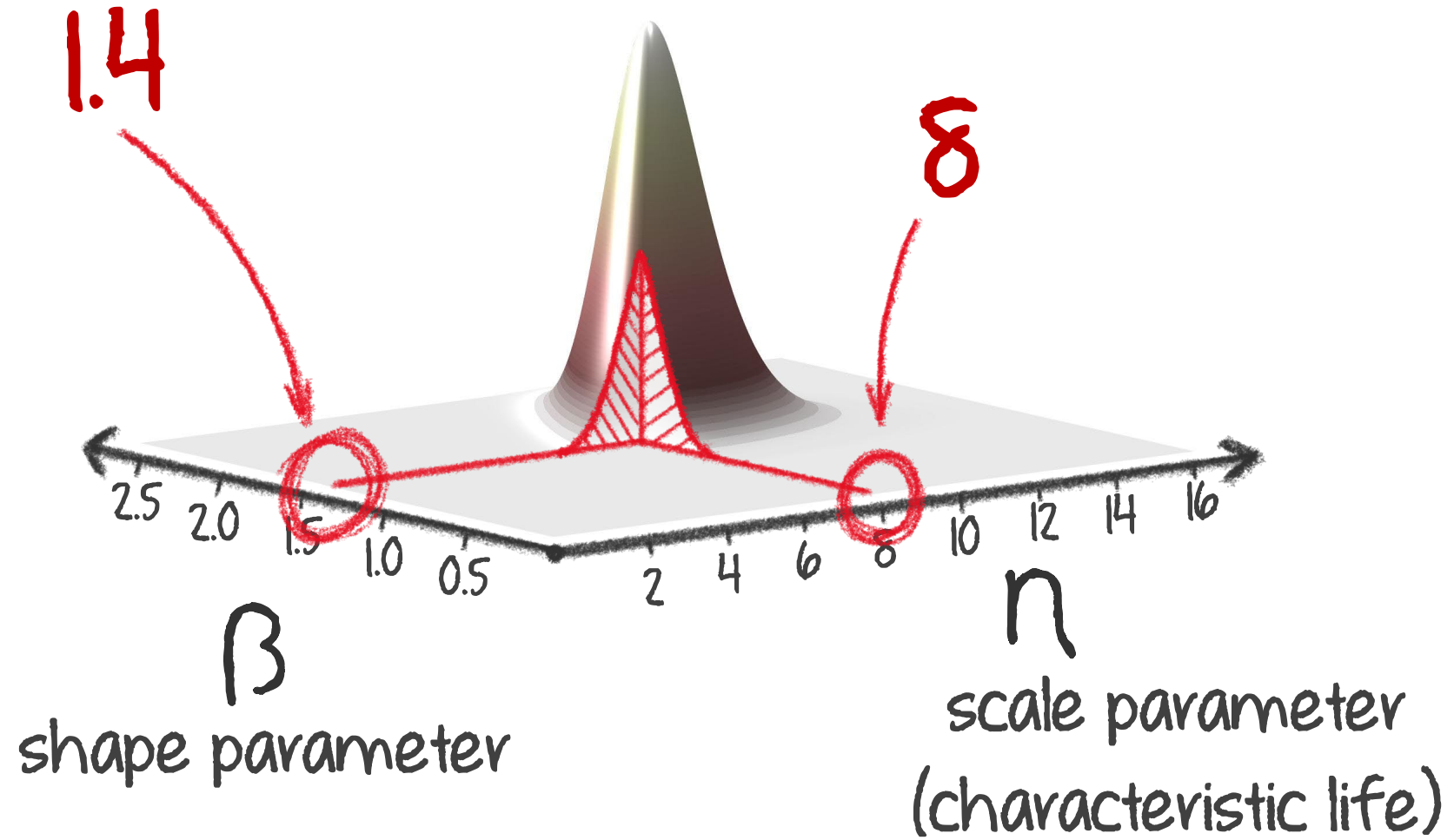
this shape



β
shape parameter

n
scale parameter
(characteristic life)



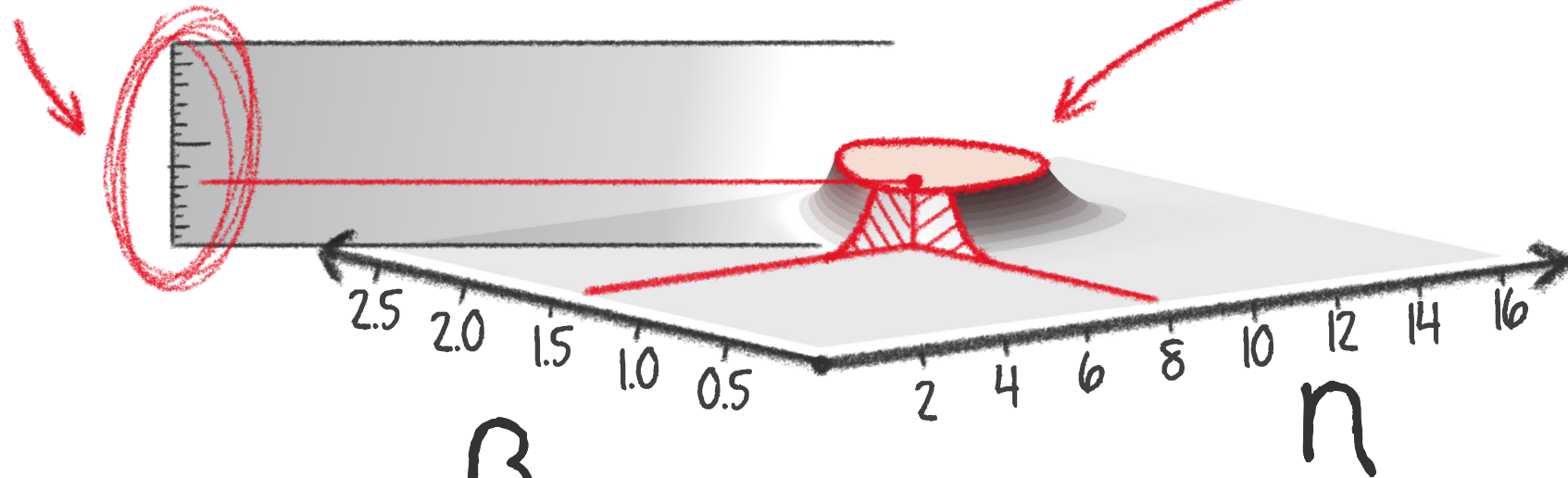


... randomly select a
value within the
likelihood

'height'
of our first point

... we use this randomly
selected value to create our

slice



β
shape parameter

n
scale parameter
(characteristic life)

... the next step is to

**randomly
select a value**

in our slice

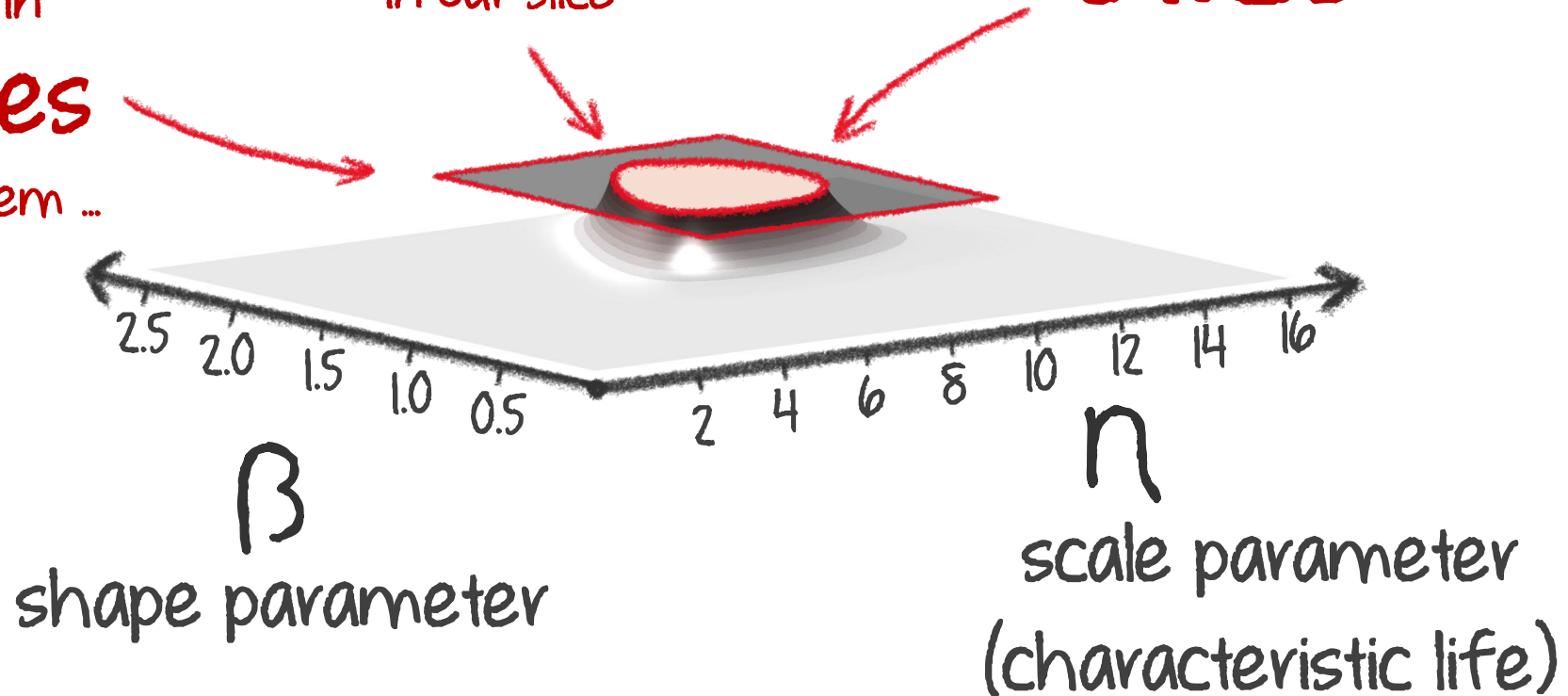
... we use this randomly
selected value to create our

slice

... but our computers
can only work in

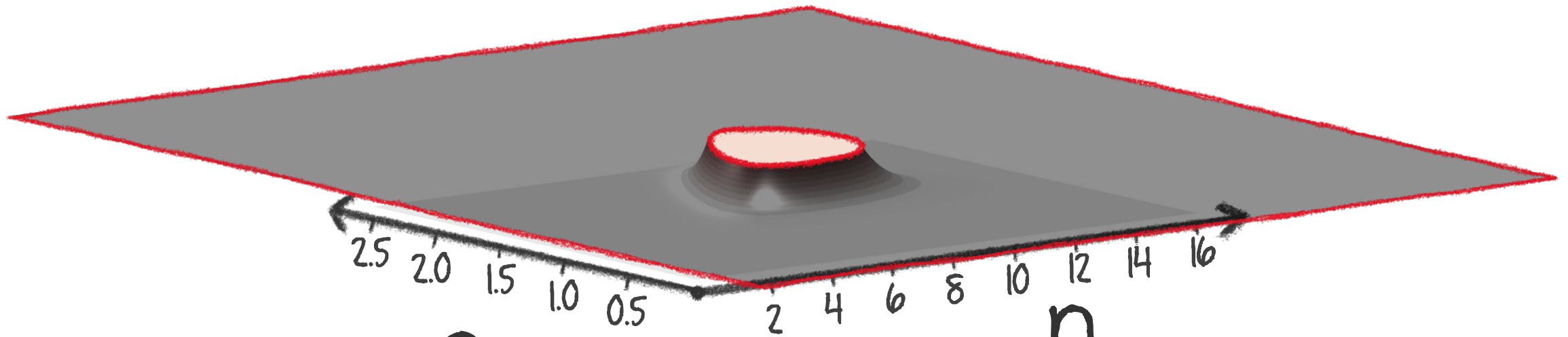
rectangles

we define for them ...



... we can help make sure
our slice is in this rectangle
by making it

big



β
shape parameter

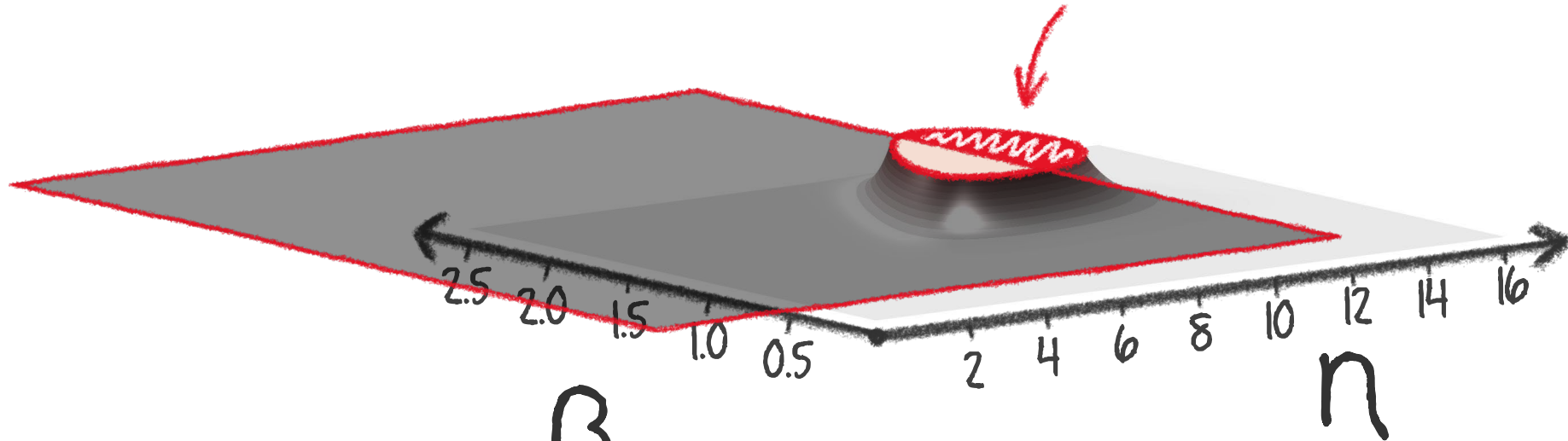
n
scale parameter
(characteristic life)

... we can help make sure
our slice is in this rectangle
by making it

big

... and we want to avoid

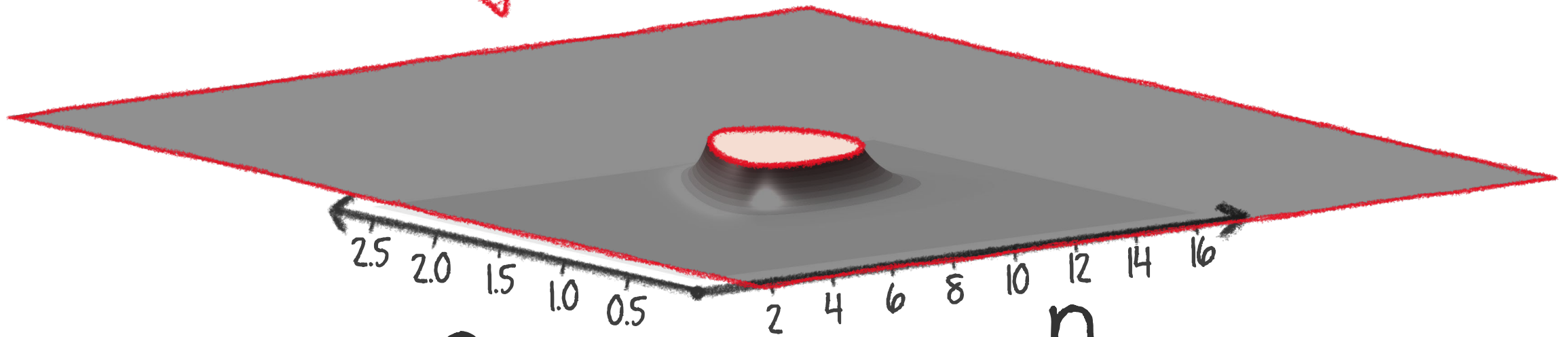
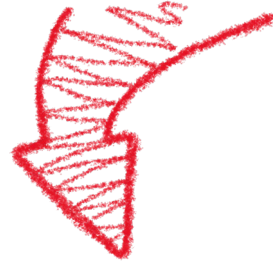
this



β
shape parameter

n
scale parameter
(characteristic life)

the rectangle of guess



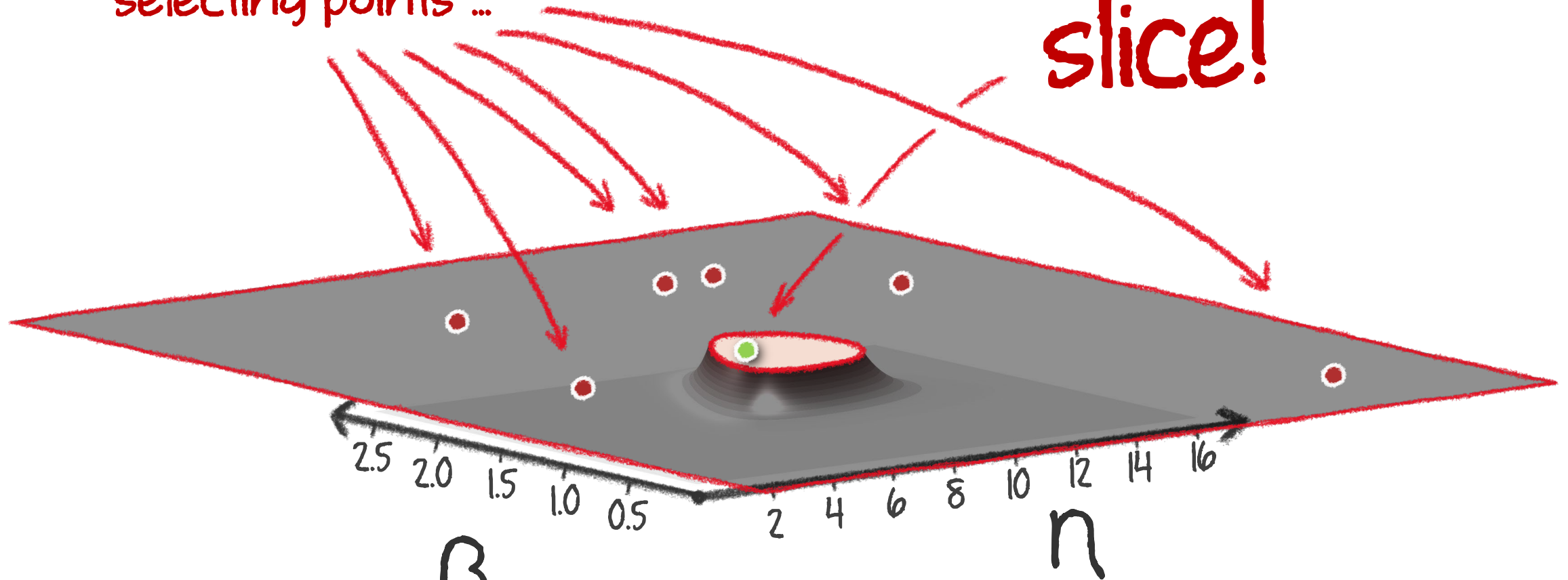
β
shape parameter

n
scale parameter
(characteristic life)

... randomly
selecting points ...

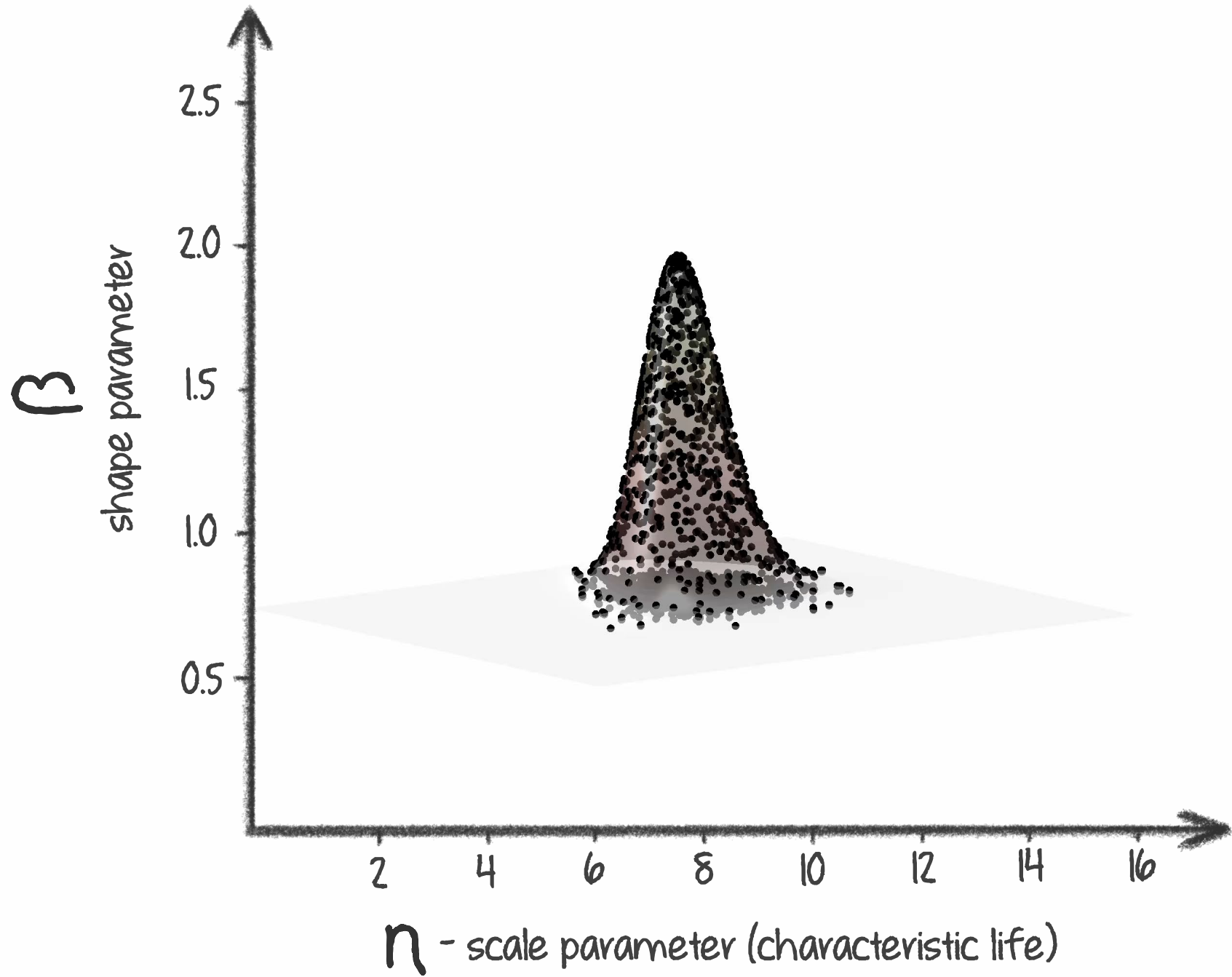
... until we find one in our

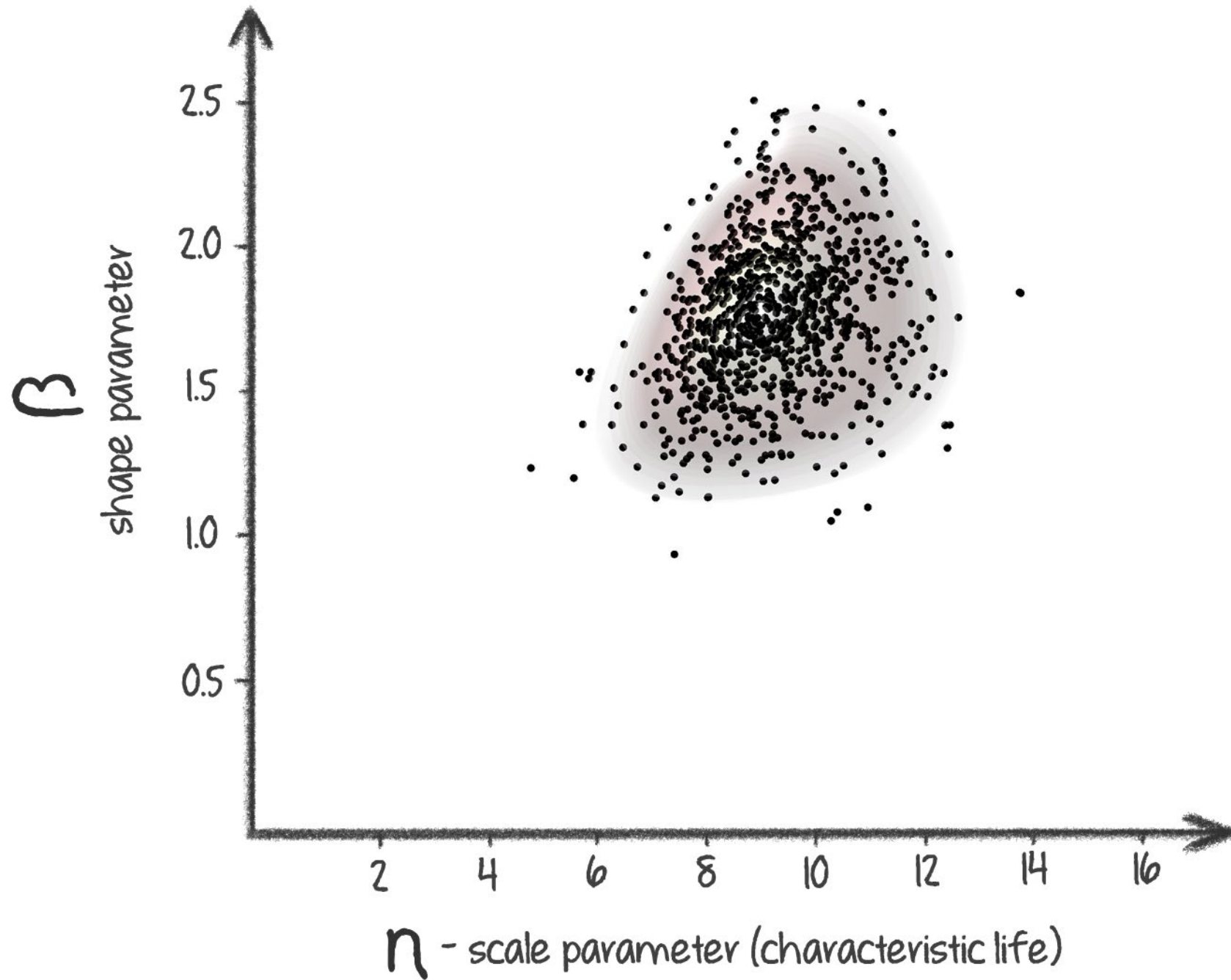
slice!



β
shape parameter

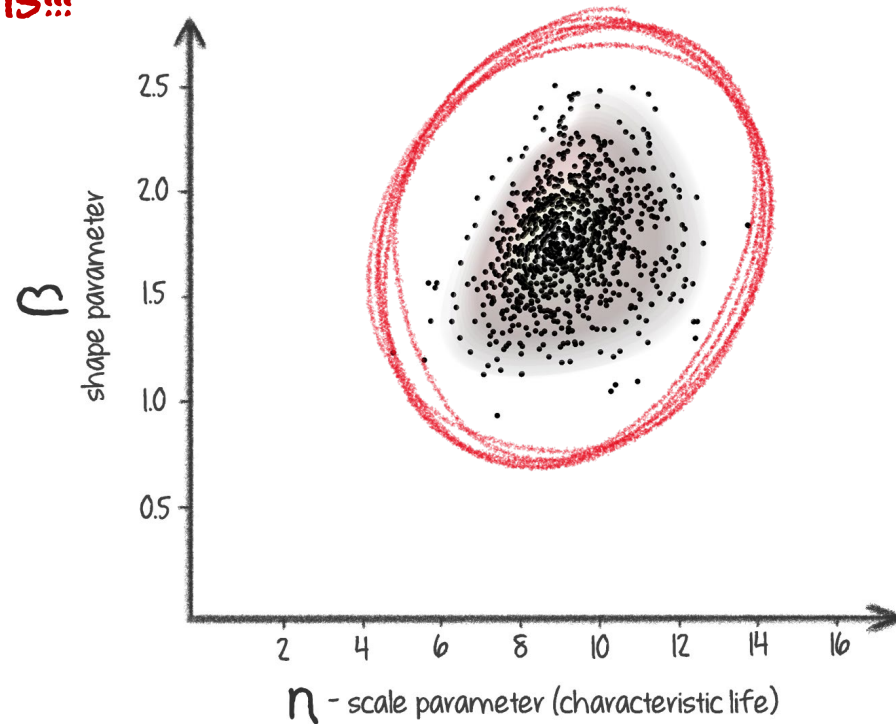
n
scale parameter
(characteristic life)



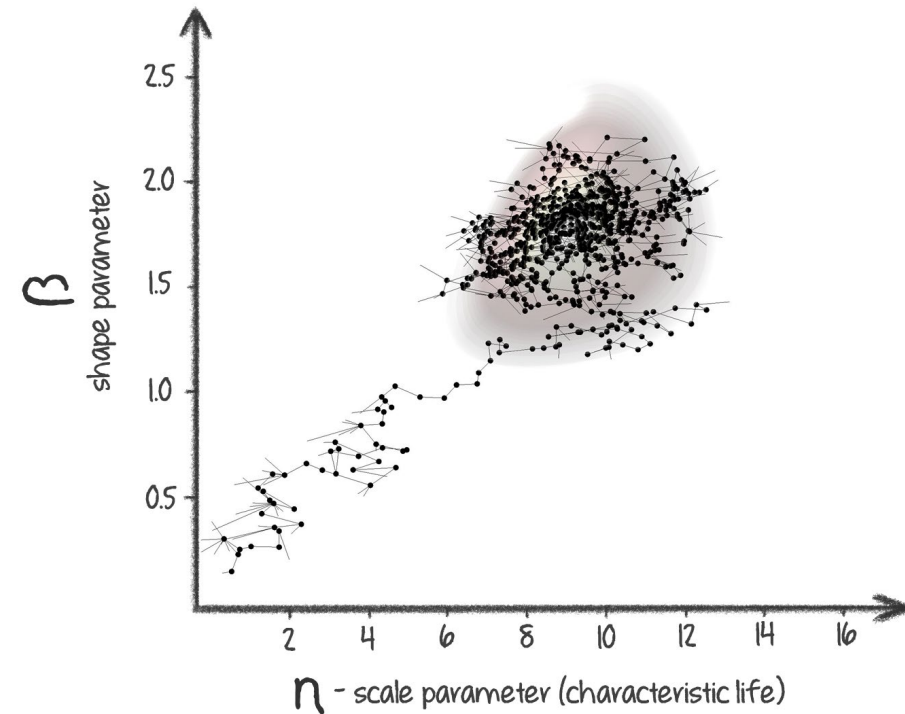


... how much
better
is this!!!

slice sampling



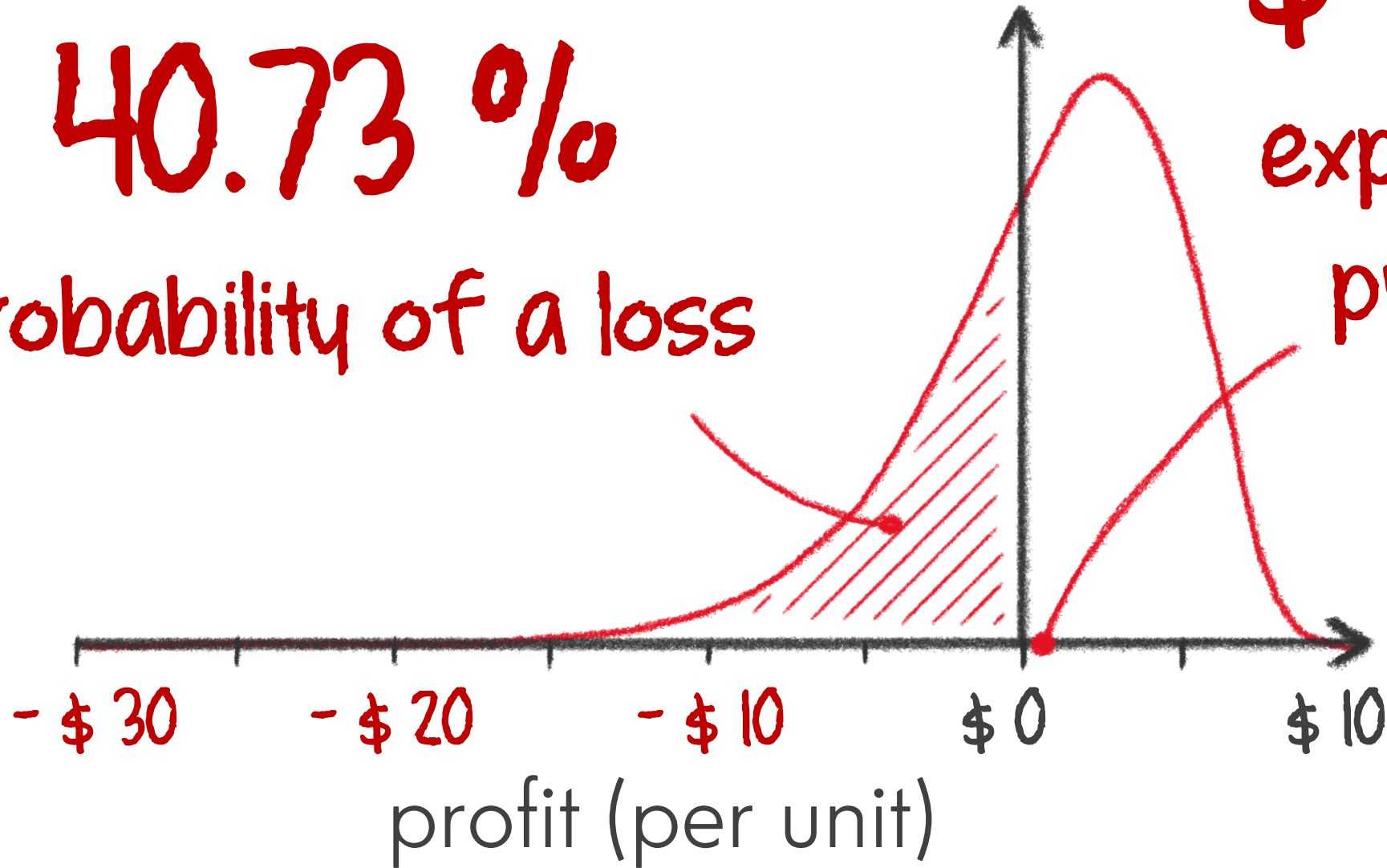
Metropolis-Hastings algorithm



... and so we have
our samples or
posse to help us
answer questions

40.73 %
probability of a loss

\$ 0.32
expected
profit



... so what
is next?

p =

slicesample ...

([1 , 1] , ...

100 , ...

'pdf' , ...

@ (p) prod(wb1pdf(x,p(1),p(2)))

... we need to create

CODE

that works

