Johnson SU distribution
MLE (Maximum Likelihood Estimate) fit to determine parameters
LR (Likelihood Ratio) approach to find tolerance limit

Pseudo-Code Used to Describe Process

Input

```
x     <- iris$Sepal.Width
sided <- 1
alpha <- 0.01  # confidence = 1 – alpha / sided
P     <- 0.99  # proportion or coverage
```

```
> x
  [1] 3.5 3.0 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 3.7 3.4 3.0 3.0 4.0 4.4 3.9 3.5 3.8 3.8
 [21] 3.4 3.7 3.6 3.3 3.4 3.0 3.4 3.5 3.4 3.2 3.1 3.4 4.1 4.2 3.1 3.2 3.5 3.6 3.0 3.4
 [41] 3.5 2.3 3.2 3.5 3.8 3.0 3.8 3.2 3.7 3.3 3.2 3.2 3.1 2.3 2.8 2.8 3.3 2.4 2.9 2.7
 [61] 2.0 3.0 2.2 2.9 2.9 3.1 3.0 2.7 2.2 2.5 3.2 2.8 2.5 2.8 2.9 3.0 2.8 3.0 2.9 2.6
 [81] 2.4 2.4 2.7 2.7 3.0 3.4 3.1 2.3 3.0 2.5 2.6 3.0 2.6 2.3 2.7 3.0 2.9 2.9 2.5 2.8
[101] 3.3 2.7 3.0 2.9 3.0 3.0 2.5 2.9 2.5 3.6 3.2 2.7 3.0 2.5 2.8 3.2 3.0 3.8 2.6 2.2
[121] 3.2 2.8 2.8 2.7 3.3 3.2 2.8 3.0 2.8 3.0 2.8 3.8 2.8 2.8 2.6 3.0 3.4 3.1 3.0 3.1
[141] 3.1 3.1 2.7 3.2 3.3 3.0 2.5 3.0 3.4 3.0
```

Determine Johnson Su Parameters

```
## fits using standard R packages for comparison
      gamma    delta         xi   lambda type              description
1  0.000000 8.225235 -0.5740271 3.587063   SL SuppDists::JohnsonFit(x)
2 -3.306477 5.319413  1.7846077 1.887725   SU   ExtDist::eJohnsonSU(x)

## Define a useful function to keep lambda well behaved during MLE optimization
lambda.fix <- function() {lambda <- abs(lambda); if (lambda == 0) lambda <- 1E-15}

## Define nll (negative log likelihood) function for MLE fit
## parameters: gamma, delta, xi, lambda
nll <- function() {
    lambda <- lambda.fix()
    z   <- (x-xi)/lambda
    pdf <- delta /( lambda * sqrt(2 * pi) ) * 1 / sqrt(1 + z^2) *
           exp( -0.5*(gamma + delta * asinh(z))^2 )
    nll <- -sum(log(pdf))
}

## MLE fit using the above to minimize nnl() and return gamma, delta, xi, and lambda
## Coded using the standard optimization module stats::optim()to find the values of
## gamma, delta, xi, and lambda that minimize nnl.
      gamma    delta         xi   lambda type              description
4 -3.306484 5.319412  1.7846186 1.887725   SU                      MLE
```

Calculate Tolerance Limit

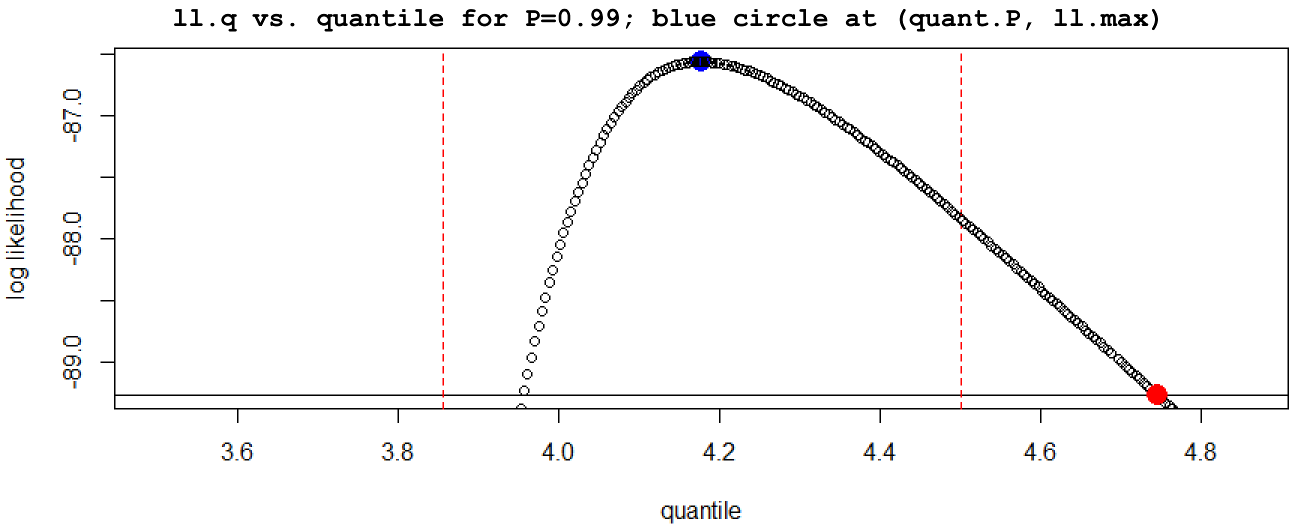
(1) Define function to calculate nll for Johnson SU but with quantile (quant) and coverage (P) as specified parameters and quant, delta, xi, and lambda as fitted parameters. Use function to find maximum log likelihood and the quantile associated with coverage, P.

```
nll.q <- function(given=c(x=data, P),
                  fit=c(quant, delta, xi, lambda),
                  return=list(nll, fit parameters) {
    lambda <- lambda.fix()
    ## write gamma as a function of quant, delta, xi and lambda
    gamma <- qnorm(P) - delta * asinh( (quant-xi)/lambda )
    ## PDF for Johnson SU
    z   <- (x-xi)/lambda
    pdf <- delta /( lambda * sqrt(2 * pi) ) * 1 / sqrt(1 + z^2) *
           exp( -0.5*(gamma + delta * asinh(z))^2 )
    nll     <- -sum(log(pdf))
```

```
}
## Peak of log likelihood function (-nll.q())for coverage, P, is at
## quantile, quant.P, and log likelihood, ll.max
quant.P = 4.178656
ll.max = -86.55915
```

(2) Define function nll.fixedq for Johnson SU but for a fixed coverage (P) and quantile (quant) with fit parameters delta, xi, and lambda.

```
nll.fixedq <- function(given=c(x=data, quant, P),
                       fit=c(delta, xi, lambda),
                       return=ll.q) {
    param <- list(quant=quant, delta=param[[1]],
                  xi=param[[2]], lambda=param[[3]])
    ll.q <- nll.q(data, param, P, lambda.control, debug=FALSE)
}
ll.q <- -nll.fixedq()
```



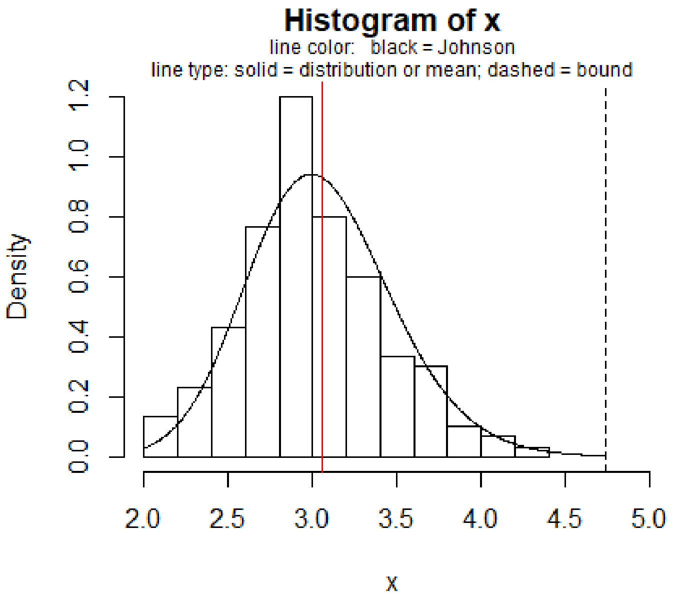
(3) Calculate Confidence Limits on Quantile Using LR (Likelihood Ratio)

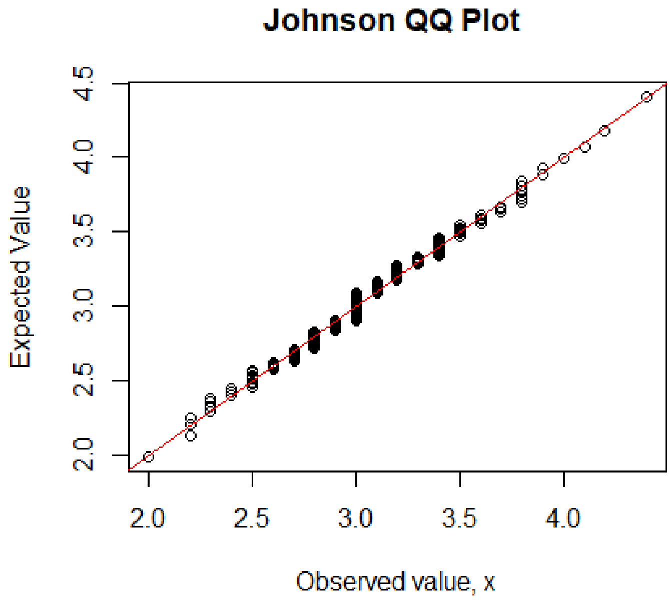
```
## Reduce peak by chi-squared to find tolerance limit.
## Factor of 2 in the following assumes alpha was specified for 1-sided, but use
## to determine the confidence portion of a tolerance limit is always 2-sided.
## The difference in a 1 or 2-sided tolerance limit comes from P, not alpha.
ll.tol <-  ll.max - qchisq(1 - 2*alpha, 1)  # qchisq(1-2*0.01, 1) = 5.411894
ll.tol = -89.2651

## Confidence limit (red circle) is at the intersection of ll.tol and the
## log likelihood function, ll.q(), for given level of coverage, P. Current
## coding uses a Newton-Raphson search to find the intersection.
tol.upper <- newton.raphson(ll.q, ll.tol)

## Based on the above:
  alpha    P sided tol.lower tol.upper
1  0.01 0.99     1        NA  4.746048
```

Histogram with Upper, 1-Sided, 99/99 Tolerance Limit and QQ Plot





Actual Coding Used to Recreate Above Output

```
## Put common packages and functions defined in folder "modules" into namespace.
## The "modules" folder contains the functions used further below.
source("setup.r")

## define input
x      <- iris$Sepal.Width

## use MLE to determine fit parameters
out.fit <- mle.johnsonsu(x, plots=TRUE)

## use MLE and LR (Likelihood Ratio) approach to determine
## upper, 1-sided, 99/99 tolerance limit
out.tol <- mle.johnsonsu.tol(x, plots=TRUE)

## create histogram with Johnson SU fit (type='j') and show tolerance limit
out.hist <- hist_nwj(x, type='j')

## create QQ plot for Johnson SU fit
out.qq   <- qqplot_nwj(x, type='j')
```

Resources

The approach described in the pseudo-code section is based on the approach found here:

- https://www.r-bloggers.com/2019/08/maximum-likelihood-estimation-from-scratch/
- https://personal.psu.edu/abs12/stat504/Lecture/lec3_4up.pdf