

Fundamentals of Statistical Modeling (VT20)

Andrea Discacciati
Karolinska Institutet
Stockholm, Sweden

Lab 1

Load the dataset and the mlci command

```
. version 14
. use https://raw.githubusercontent.com/anddis/fsm/master/data/lab1.dta, clear
. run https://raw.githubusercontent.com/anddis/fsm/master/do/mlci.do
```

Exercise 0 (together to get started)

We assume that $f(y_n)$ is normal. Estimate the parameters μ and σ .

```
. local f = "exp(-(y_n-{mu})^2/(2*{sigma}^2))/sqrt(2*_pi*{sigma}^2)"
. mlexp (ln(`f`))
initial:      log likelihood =      -<inf>   (could not be evaluated)
feasible:      log likelihood = -43652.817
rescale:      log likelihood = -1800.6375
rescale eq:    log likelihood = -1328.2447
Iteration 0:   log likelihood = -1328.2447   (not concave)
Iteration 1:   log likelihood = -1120.8309
Iteration 2:   log likelihood = -1064.5796
Iteration 3:   log likelihood = -1060.4702
Iteration 4:   log likelihood = -1059.3325
Iteration 5:   log likelihood = -1059.3319
Iteration 6:   log likelihood = -1059.3319
Maximum likelihood estimation
Log likelihood = -1059.3319                Number of obs   =           300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
/mu	178.4911	.4772459	374.00	0.000	177.5557	179.4265
/sigma	8.266142	.3374638	24.49	0.000	7.604725	8.927559

```
.
. di "The MLE for mu is: "_b[/mu]
The MLE for mu is: 178.49108
. di "The MLE for sigma is: "_b[/sigma]
The MLE for sigma is: 8.2661417
```

Exercise 1

Some of the distributions we'll work with in the course/labs are:

- Normal: https://en.wikipedia.org/wiki/Normal_distribution
- Skew-normal: https://en.wikipedia.org/wiki/Skew_normal_distribution
- Gamma: https://en.wikipedia.org/wiki/Gamma_distribution
- Beta: https://en.wikipedia.org/wiki/Beta_distribution
- Exponential: https://en.wikipedia.org/wiki/Exponential_distribution
- Chi-squared: https://en.wikipedia.org/wiki/Chi-squared_distribution
- Weibull: https://en.wikipedia.org/wiki/Weibull_distribution

- Bernoulli: https://en.wikipedia.org/wiki/Bernoulli_distribution
- Negative binomial: https://en.wikipedia.org/wiki/Negative_binomial_distribution

Get familiar with the distributions above.

- Are they useful to model continuous or discrete variables?
- What's their support?
- By how many parameters are they parametrized?
- What does their shape look like?

Exercise 2

We assume that $f(y_n)$ is normal. Estimate the parameters μ and σ . Use the `normalden()` function.

```
. local f = "normalden(y_n, {mu}, {sigma})"
. mlexp (ln(`f`))
```

```
initial:      log likelihood =      -<inf>  (could not be evaluated)
feasible:      log likelihood = -43652.817
rescale:      log likelihood = -1800.6375
rescale eq:    log likelihood = -1328.2447
Iteration 0:    log likelihood = -1328.2447  (not concave)
Iteration 1:    log likelihood = -1120.8299
Iteration 2:    log likelihood = -1064.4927
Iteration 3:    log likelihood = -1060.4292
Iteration 4:    log likelihood = -1059.3324
Iteration 5:    log likelihood = -1059.3319
Iteration 6:    log likelihood = -1059.3319
```

Maximum likelihood estimation

Log likelihood = -1059.3319	Number of obs	=	300
-----------------------------	---------------	---	-----

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/mu	178.4911	.4772459	374.00	0.000	177.5557 179.4265
/sigma	8.266142	.3374638	24.49	0.000	7.604725 8.927559

Estimate again the parameters μ and σ , but this time constrain the parameter σ to be positive by replacing $\sigma = \exp(\theta)$. You'll get now an estimate of θ . Recover the MLE of σ . (Note that the MLE $\hat{\sigma}$ hasn't changed – see slide 33 – but this “trick” improves computational stability).

```
. local sigma = "exp({theta})"
. local f = "normalden(y_n, {mu}, `sigma`)"
. mlexp(ln(`f`))
```

```
initial:      log likelihood =      -<inf>  (could not be evaluated)
feasible:      log likelihood = -32398.765
rescale:      log likelihood = -1981.1218
rescale eq:    log likelihood = -1440.3171
Iteration 0:    log likelihood = -1440.3171  (not concave)
Iteration 1:    log likelihood = -1112.6119
Iteration 2:    log likelihood = -1085.7986
Iteration 3:    log likelihood = -1059.4172
Iteration 4:    log likelihood = -1059.332
Iteration 5:    log likelihood = -1059.3319
```

Maximum likelihood estimation

Log likelihood = -1059.3319	Number of obs	=	300
-----------------------------	---------------	---	-----

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/mu	178.4911	.4772459	374.00	0.000	177.5557 179.4265
/theta	2.112168	.0408248	51.74	0.000	2.032153 2.192183

```
.
. di exp(_b[/theta])
8.2661406
```

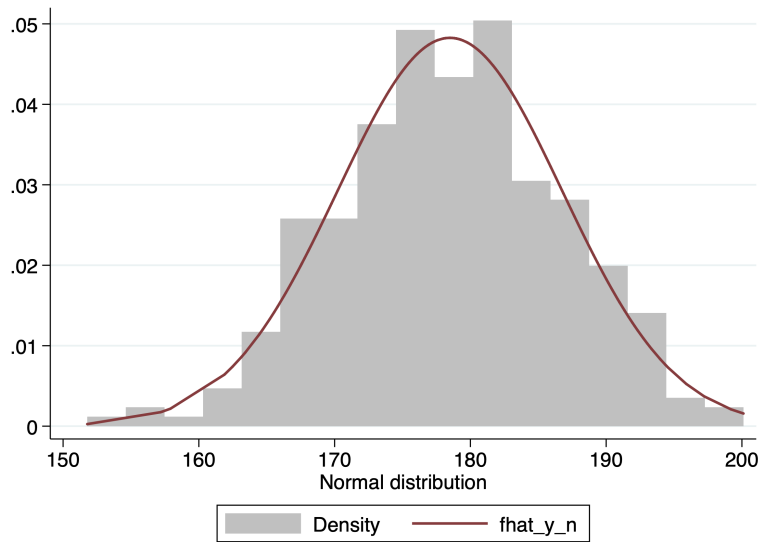
Plot the estimated density $\hat{f}(y_n)$ over the sample histogram.

```
. gen fhat_y_n = normalden(y_n, _b[/mu], exp(_b[/theta]))
```

```

. tw (hist y_n) (line fhat_y_n y_n, sort), name(y_n, replace)
. graph export y_n.png, replace
(file y_n.png written in PNG format)

```

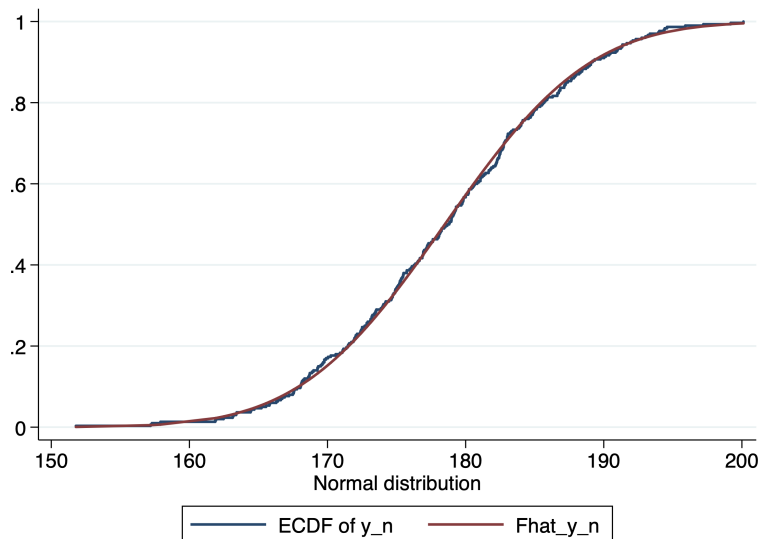


Plot the model-based estimated CDF $\hat{F}(y_n)$ (see slide 38) over the empirical CDF (see slide 20). The function `normal()` returns the CDF of a standard normal distribution.

```

. gen Fhat_y_n = normal((y_n - _b[/mu])/exp(_b[/theta]))
. cumul y_n, gen(sampleF_y_n)
.
. // This is what -cumul- does under the hood
. // sort y_n
. // gen sampleF_y_n = sum(_cons)
. // su sampleF_y_n
. // replace sampleF_y_n = sampleF_y_n / r(max)
.
. tw (line sampleF_y_n Fhat_y_n y_n, connect(J 1) sort), name(Fy_n, replace)
. graph export Fy_n.png, replace
(file Fy_n.png written in PNG format)

```



Exercise 3

Assume that $f(y_g)$ is (2-parameter) gamma. Estimate the parameters α and β using the `gammaden()` function. Fix the location parameter g (the third argument of the `gammaden()` function), equal to 0.

```
. local f = "gammaden({alpha}, {beta}, 0, y_g)"
. mlexp(ln(`f`))
```

```
initial:      log likelihood =      -<inf>   (could not be evaluated)
feasible:      log likelihood = -2465.2095
rescale:      log likelihood = -670.37304
rescale eq:   log likelihood = -670.37304
Iteration 0:   log likelihood = -670.37304
Iteration 1:   log likelihood = -668.66319
Iteration 2:   log likelihood = -668.35073
Iteration 3:   log likelihood = -668.34648
Iteration 4:   log likelihood = -668.34648
```

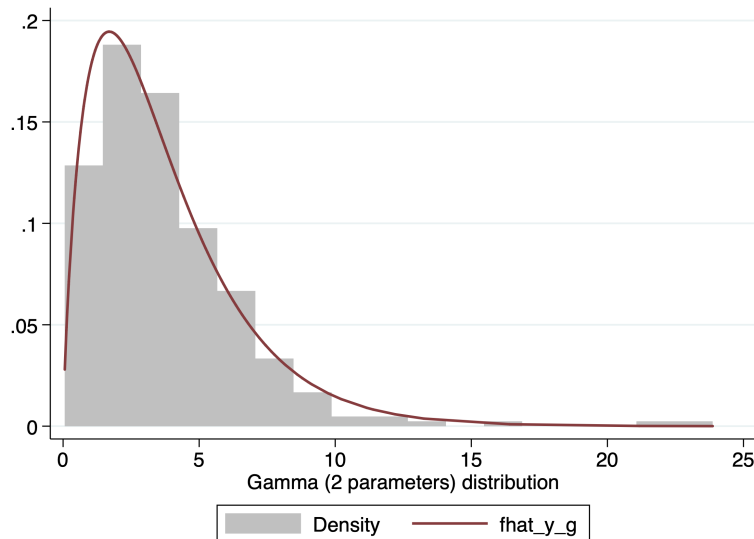
Maximum likelihood estimation

```
Log likelihood = -668.34648                Number of obs   =          300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/alpha	1.830878	.138033	13.26	0.000	1.560338 2.101418
/beta	2.042909	.1769745	11.54	0.000	1.696046 2.389773

Plot the estimated density $\hat{f}(y_g)$ over the sample histogram.

```
. gen fhat_y_g = gammaden(_b[/alpha], _b[/beta], 0, y_g)
. tw (hist y_g) (line fhat_y_g y_g, sort), name(y_g, replace)
. graph export y_g.png, replace
(file y_g.png written in PNG format)
```



Exercise 4

We assume that $f(y_b)$ is beta. Estimate the parameters α and β using the `betaden()` function.

```
. local f = "betaden({alpha}, {beta}, y_b)"
. mlexp(ln(`f`))
```

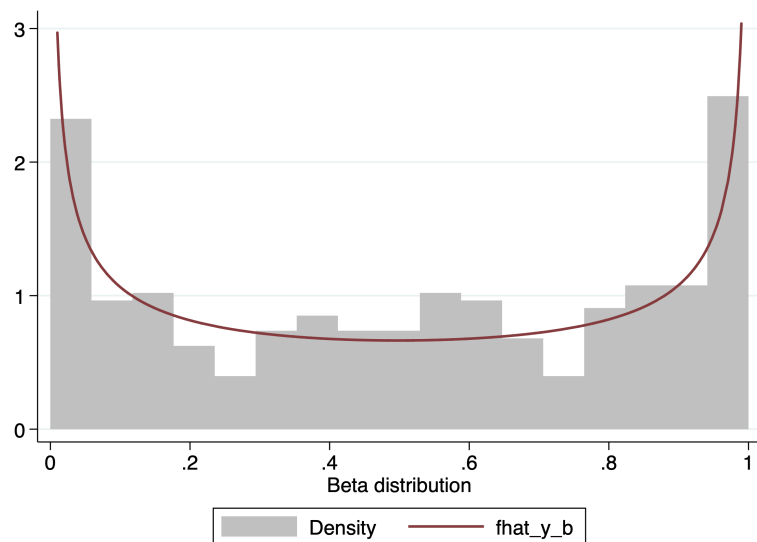
```
initial:      log likelihood =      -<inf>   (could not be evaluated)
feasible:      log likelihood = 57.883521
rescale:      log likelihood = 57.883521
rescale eq:   log likelihood = 57.883521
Iteration 0:   log likelihood = 57.883521
Iteration 1:   log likelihood = 58.345242
Iteration 2:   log likelihood = 58.346286
```

```
Iteration 3:  log likelihood =  58.346286
Maximum likelihood estimation
Log likelihood =  58.346286          Number of obs   =        300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
/alpha	.5346582	.0394503	13.55	0.000	.457337	.6119794
/beta	.5286525	.0388869	13.59	0.000	.4524355	.6048694

Plot the estimated density $\hat{f}(y_b)$ over the sample histogram. (OBS: We need to be careful when plotting the estimated density function close to the boundaries of the distribution's support).

```
. gen fhat_y_b = betaden(_b[/alpha], _b[/beta], y_b)
. tw (hist y_b) (line fhat_y_b y_b if inrange(y_b, .01, .99), sort), name(y_b, replace)
. graph export y_b.png, replace
(file y_b.png written in PNG format)
```



Exercise 5

We assume that $f(y_e)$ is exponential. Estimate the parameter λ . You'll need to code the density of an exponential distribution yourself: $f(y; \lambda) = \lambda \exp(-\lambda y)$

```
. local f = "{lambda}*exp(-y_e * {lambda})"
. mlexp(ln(`f`))
initial:      log likelihood =      -<inf>  (could not be evaluated)
feasible:      log likelihood =   -308.3512
rescale:      log likelihood =  -193.68405
Iteration 0:    log likelihood =  -193.68405
Iteration 1:    log likelihood =  -181.67541
Iteration 2:    log likelihood =  -179.57917
Iteration 3:    log likelihood =  -179.57914
Iteration 4:    log likelihood =  -179.57914
Maximum likelihood estimation
Log likelihood =  -179.57914          Number of obs   =        300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
/lambda	1.493919	.0862515	17.32	0.000	1.324869	1.662969

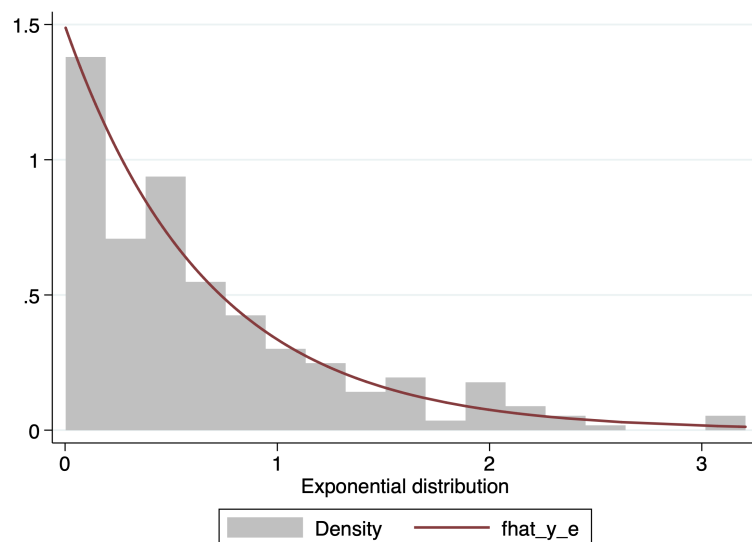
Plot the estimated density $\hat{f}(y_e)$ over the sample histogram.

```
. gen fhat_y_e = _b[/lambda] * exp(-y_e * _b[/lambda])
```

```

. tw (hist y_e) (line fhat_y_e y_e, sort), name(y_e, replace)
. graph export y_e.png, replace
(file y_e.png written in PNG format)

```



Exercise 6

We assume that $f(y_c)$ is chi-squared. Estimate the parameter k using the `chi2den()` function.

```

. local f = "chi2den({k}, y_c)"
. mlexp(ln(`f`))
initial:      log likelihood =      -<inf>  (could not be evaluated)
feasible:      log likelihood = -1053.9877
rescale:      log likelihood = -660.78037
Iteration 0:    log likelihood = -660.78037
Iteration 1:    log likelihood = -628.29724
Iteration 2:    log likelihood = -626.40531
Iteration 3:    log likelihood = -626.40048
Iteration 4:    log likelihood = -626.40048
Maximum likelihood estimation
Log likelihood = -626.40048          Number of obs   =       300

```

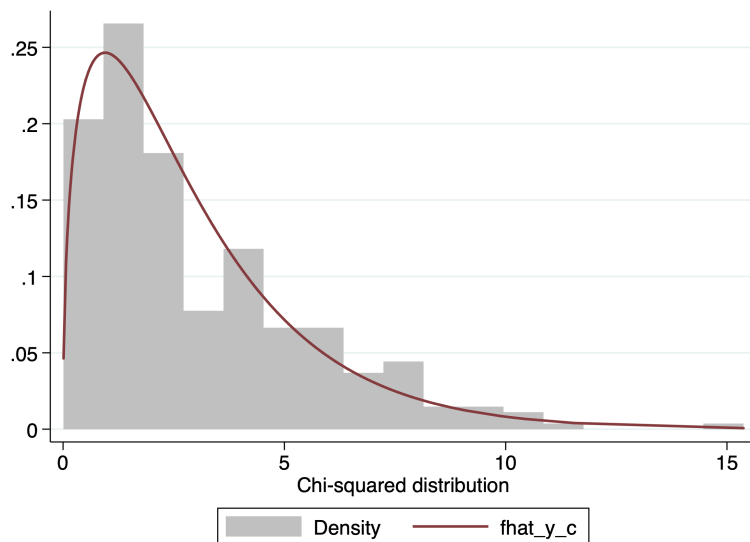
	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/k	2.950201	.1181046	24.98	0.000	2.718721 3.181682

Plot the estimated density $\hat{f}(y_c)$ over the sample histogram.

```

. gen fhat_y_c = chi2den(_b[/k], y_c)
. tw (hist y_c) (line fhat_y_c y_c, sort), name(y_c, replace)
. graph export y_c.png, replace
(file y_c.png written in PNG format)

```



Exercise 7

Install the `qplot` command (you need to be connected to the Internet)

```
. net sj 16-3 gr42_7
```

```
package gr42_7 from http://www.stata-journal.com/software/sj16-3
```

```
TITLE
    SJ16-3 gr42_7. Update: Quantile plots
DESCRIPTION/AUTHOR(S)
    Update: Quantile plots
    by Nicholas J. Cox, Durham University,
        Department of Geography, Durham, UK
    Support:  n.j.cox@durham.ac.uk
    After installation, type help qplot
INSTALLATION FILES                                     (type net install gr42_7)
    gr42_7/qplot.ado
    gr42_7/qplot.sthlp
```

```
. net install gr42_7
checking gr42_7 consistency and verifying not already installed...
all files already exist and are up to date.
```

Let's go back to the normal distributed variable (Exercise 1). Assume that $f(y_n)$ is normal and estimate the parameters μ and σ . Generate the transform $u = \hat{F}(y_n)$ as seen during the lecture this morning. Draw the estimated quantile plot using the `qplot` command.

```
. local f = "normalden(y_n, {mu}, exp({theta}))"
. mlexp(ln(`f`))

initial:    log likelihood =    -<inf>  (could not be evaluated)
feasible:    log likelihood = -32398.765
rescale:    log likelihood = -1981.1218
rescale eq:  log likelihood = -1440.3171
Iteration 0:  log likelihood = -1440.3171  (not concave)
Iteration 1:  log likelihood = -1112.6119
Iteration 2:  log likelihood = -1085.7986
Iteration 3:  log likelihood = -1059.4172
Iteration 4:  log likelihood = -1059.332
Iteration 5:  log likelihood = -1059.3319

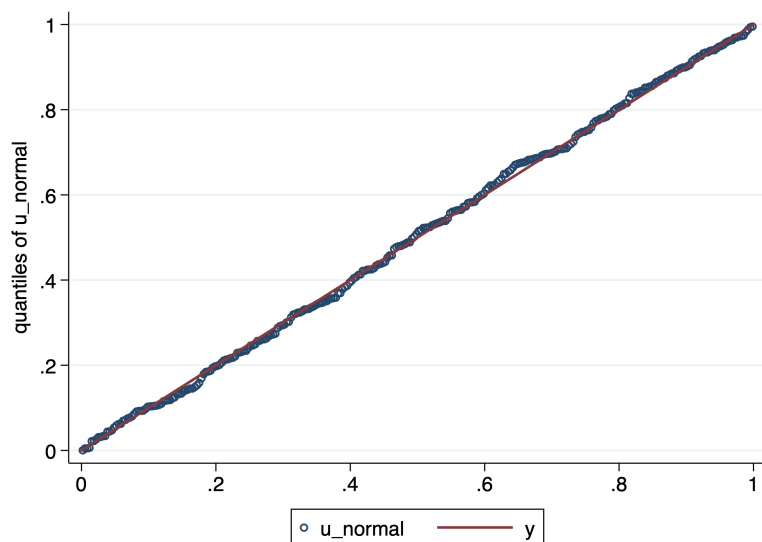
Maximum likelihood estimation
Log likelihood = -1059.3319                Number of obs   =       300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]

/mu	178.4911	.4772459	374.00	0.000	177.5557	179.4265
/theta	2.112168	.0408248	51.74	0.000	2.032153	2.192183

```
. mlci exp /theta
8.266141 95% CI: 7.630494, 8.954739

.
. gen u_normal = normal((y_n-_b[/mu])/exp(_b[/theta]))
. qplot u_normal, addplot(function y = x) name(p1, replace)
. graph export p1.png, replace
(file p1.png written in PNG format)
```



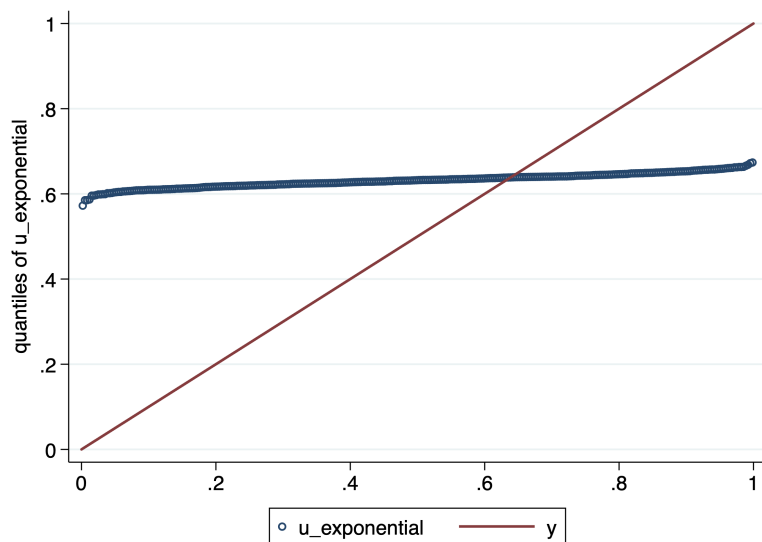
Assume now that $f(y_n)$ is exponential and estimate the parameter λ . Generate the transform $u = \hat{F}(y_n)$. Draw the estimated quantile plot using the `qplot` command.

```
. local f = "exp({theta})*exp(-y_n * exp({theta}))"
. mlexp(ln(`f`))
initial:      log likelihood = -53547.324
alternative:  log likelihood = -32628.094
rescale:      log likelihood = -2180.7534
Iteration 0:  log likelihood = -2180.7534
Iteration 1:  log likelihood = -1857.2088
Iteration 2:  log likelihood = -1855.3682
Iteration 3:  log likelihood = -1855.3616
Iteration 4:  log likelihood = -1855.3616
Maximum likelihood estimation
Log likelihood = -1855.3616          Number of obs   =       300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/theta	-5.184539	.057735	-89.80	0.000	-5.297697 -5.07138

```
. mlci exp /theta
.0056025 95% CI: .0050031, .0062738

.
. gen u_exponential = 1-exp(-y_n * exp(_b[/theta]))
. qplot u_exponential, addplot(function y = x) name(p2, replace)
. graph export p2.png, replace
(file p2.png written in PNG format)
```

Comment the two quantile plots.

Exercise 8

Assume that y_{ber} follows a Bernoulli distribution. We want to estimate the probability of “success” ($y_{ber} = 1$). Estimate the probability η while constraining it to be bounded between 0 and 1. First, write down the likelihood by hand. Then, use the `binomialp()` function.

Are the results you obtain identical to those obtained from logistic regression?

```
. local eta = "invlogit({theta})"
. local f = "`eta'^y_ber * (1-`eta')^(1-y_ber)"
. mlexp (ln(`f'))
initial:      log likelihood = -207.94415
alternative:  log likelihood = -199.7231
rescale:      log likelihood = -199.7231
Iteration 0:  log likelihood = -199.7231
Iteration 1:  log likelihood = -199.70172
Iteration 2:  log likelihood = -199.70172
Maximum likelihood estimation
Log likelihood = -199.70172          Number of obs   =       300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]
/theta	.4754237	.1187479	4.00	0.000	.2426821 .7081653

```
. mlci invlogit /theta
.6166667 95% CI: .5603745, .6699956
.
. local eta = "invlogit({theta})"
. local f = "binomialp(1, y_ber, `eta')"
```

```
. mlexp (ln(`f'))
initial:      log likelihood = -207.94415
alternative:  log likelihood = -199.7231
rescale:      log likelihood = -199.7231
Iteration 0:  log likelihood = -199.7231
Iteration 1:  log likelihood = -199.70172
Iteration 2:  log likelihood = -199.70172
Maximum likelihood estimation
Log likelihood = -199.70172          Number of obs   =       300
```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]

/theta	.4754237	.1187479	4.00	0.000	.2426821	.7081653
--------	----------	----------	------	-------	----------	----------

```

. mlci invlogit /theta
.6166667 95% CI: .5603745, .6699956
.
. logit y_ber
Iteration 0: log likelihood = -199.70172
Iteration 1: log likelihood = -199.70172
Logistic regression
Log likelihood = -199.70172
Number of obs = 300
LR chi2(0) = 0.00
Prob > chi2 = .
Pseudo R2 = 0.0000

```

y_ber	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
_cons	.4754237	.1187479	4.00	0.000	.2426821	.7081653

Now simplify the likelihood as shown in the slides. Check, again, that the estimate of η is the same.

```

. local eta = "invlogit({theta})"
. mlexp (y_ber*ln(`eta`)+(1-y_ber)*log(1-`eta`))
initial: log likelihood = -207.94415
alternative: log likelihood = -199.7231
rescale: log likelihood = -199.7231
Iteration 0: log likelihood = -199.7231
Iteration 1: log likelihood = -199.70172
Iteration 2: log likelihood = -199.70172
Maximum likelihood estimation
Log likelihood = -199.70172
Number of obs = 300

```

	Coef.	Std. Err.	z	P> z	[95% Conf. Interval]	
/theta	.4754237	.1187479	4.00	0.000	.2426821	.7081653