Bivariate probit

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \beta_2 x + \epsilon_2$

$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$

$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

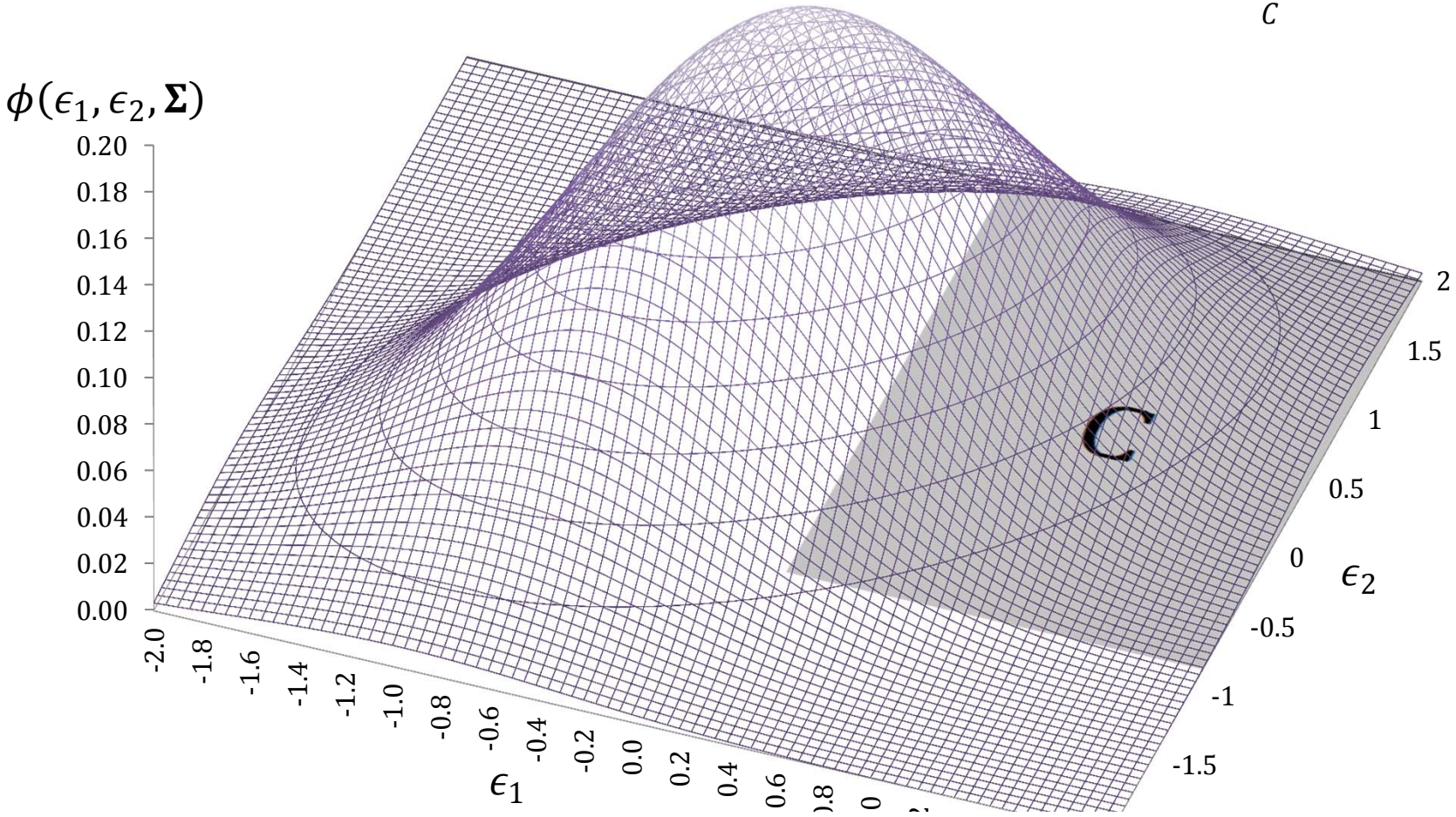
$\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$

Case $\mathbf{y} = (1,1)'$

$\epsilon_1 > -\beta_1 x, \epsilon_2 > -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 > -\beta_1 x, \epsilon_2 > -\beta_2 x\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Bivariate probit

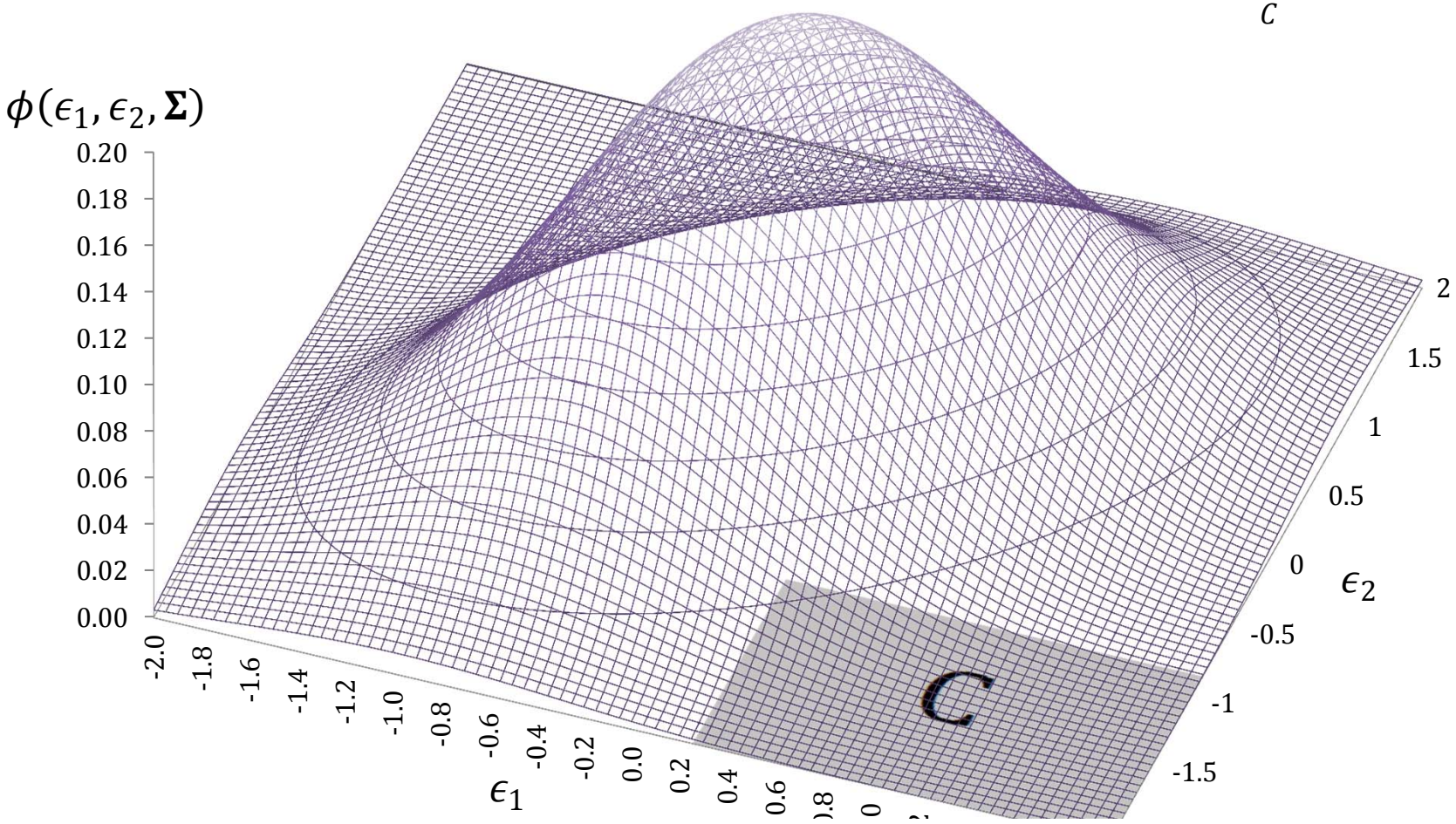
$y_1^* = \beta_1 x + \epsilon_1$
$y_2^* = \beta_2 x + \epsilon_2$
$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (1,0)'$

$\epsilon_1 > -\beta_1 x, \epsilon_2 < -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 > -\beta_1 x, \epsilon_2 < -\beta_2 x\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Bivariate probit

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \beta_2 x + \epsilon_2$

$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$

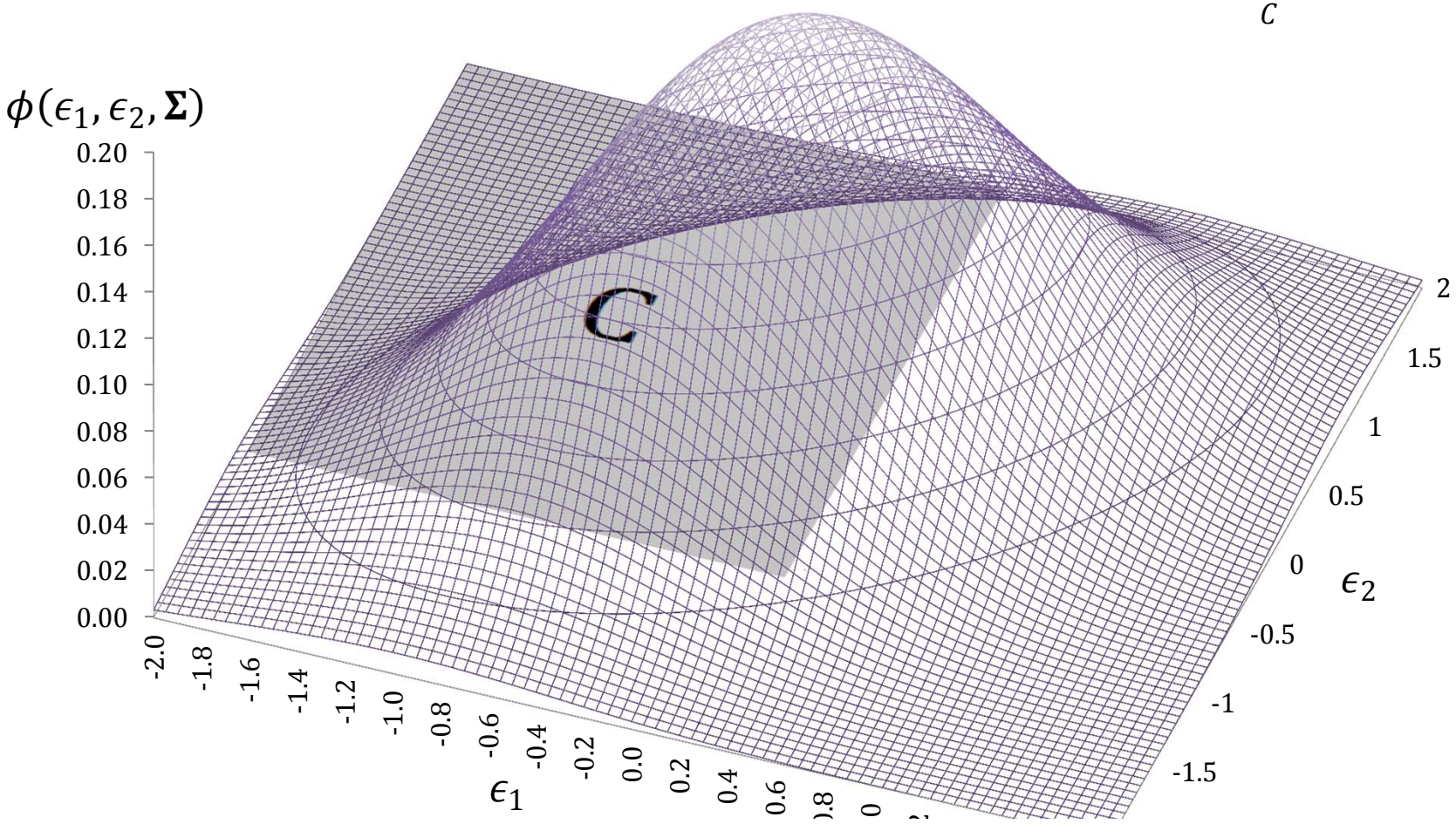
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (0,1)'$

$\epsilon_1 < -\beta_1 x, \epsilon_2 > -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 < -\beta_1 x, \epsilon_2 > -\beta_2 x\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Bivariate probit

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \beta_2 x + \epsilon_2$

$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$

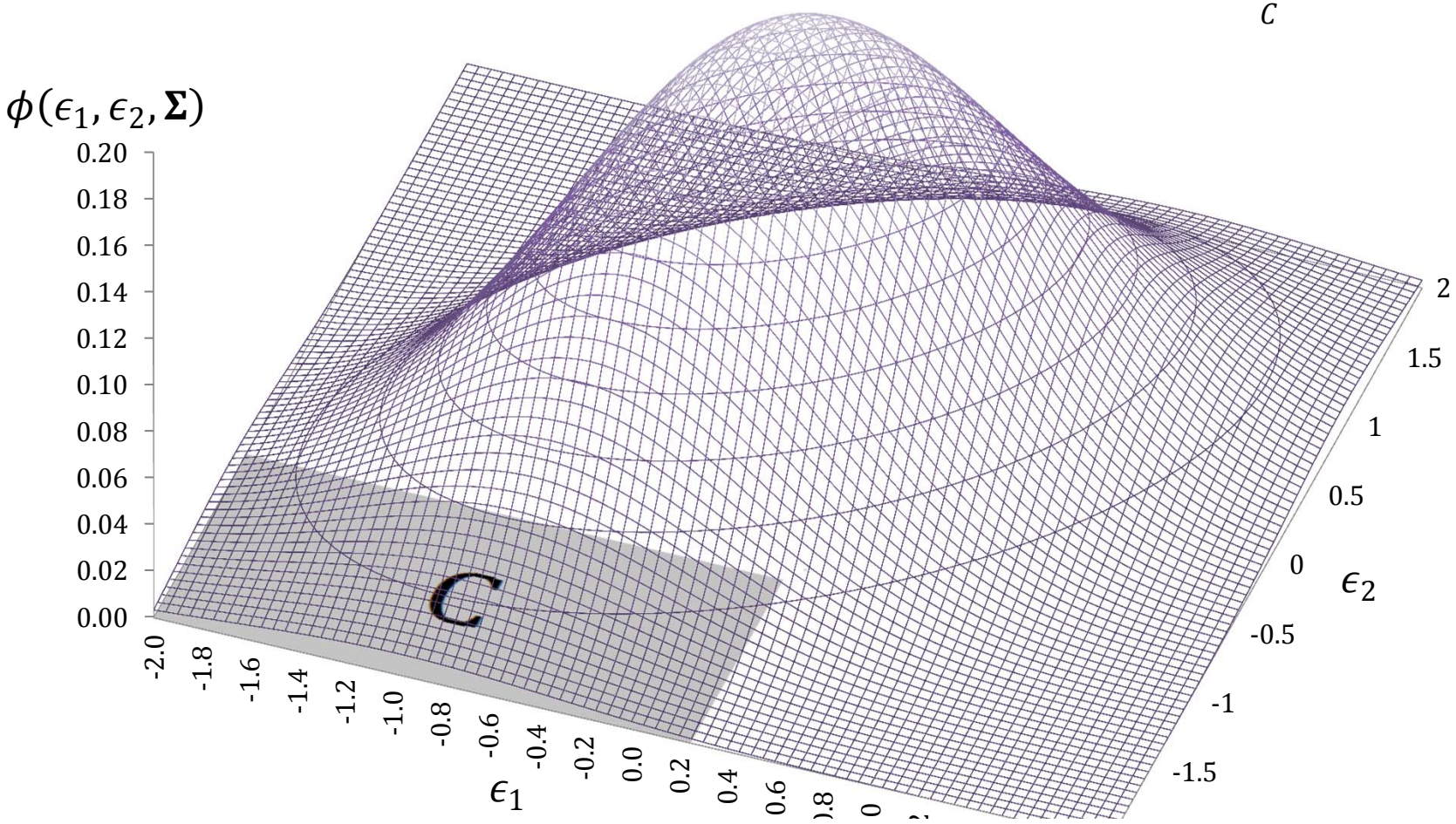
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (0,0)'$

$\epsilon_1 < -\beta_1 x, \epsilon_2 < -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 < -\beta_1 x, \epsilon_2 < -\beta_2 x\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Mixed probit-uncensored

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \beta_2 x + \epsilon_2$

$\mathbf{y} = (y_1^*, \mathbf{1}\{y_2^* > 0\})'$

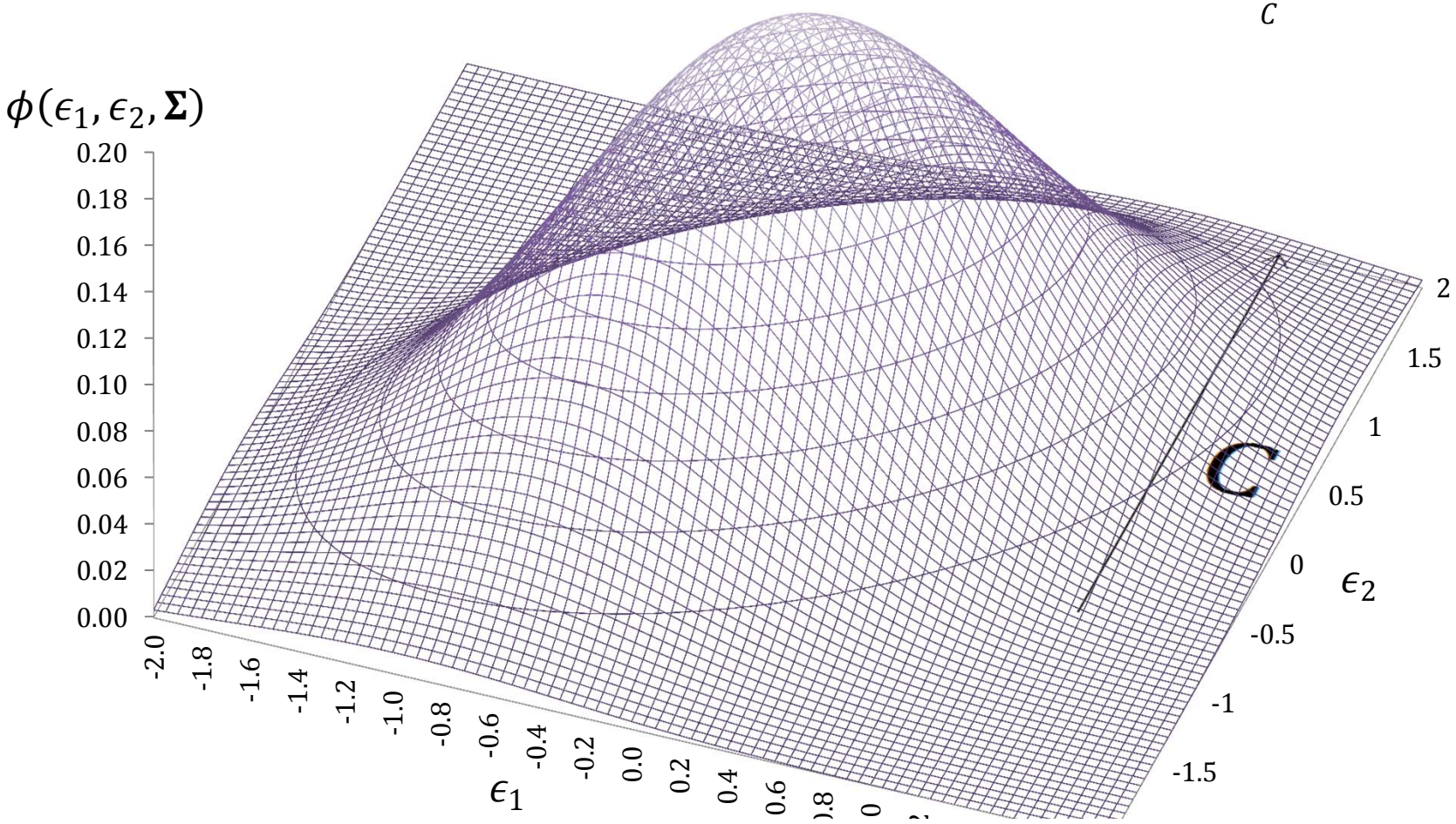
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (y_1, 1)'$

$\epsilon_1 = y_1 - \beta_1 x, \epsilon_2 > -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 = y_1 - \beta_1 x, \epsilon_2 > -\beta_2 x\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Mixed probit-uncensored

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \beta_2 x + \epsilon_2$

$\mathbf{y} = (y_1^*, \mathbf{1}\{y_2^* > 0\})'$

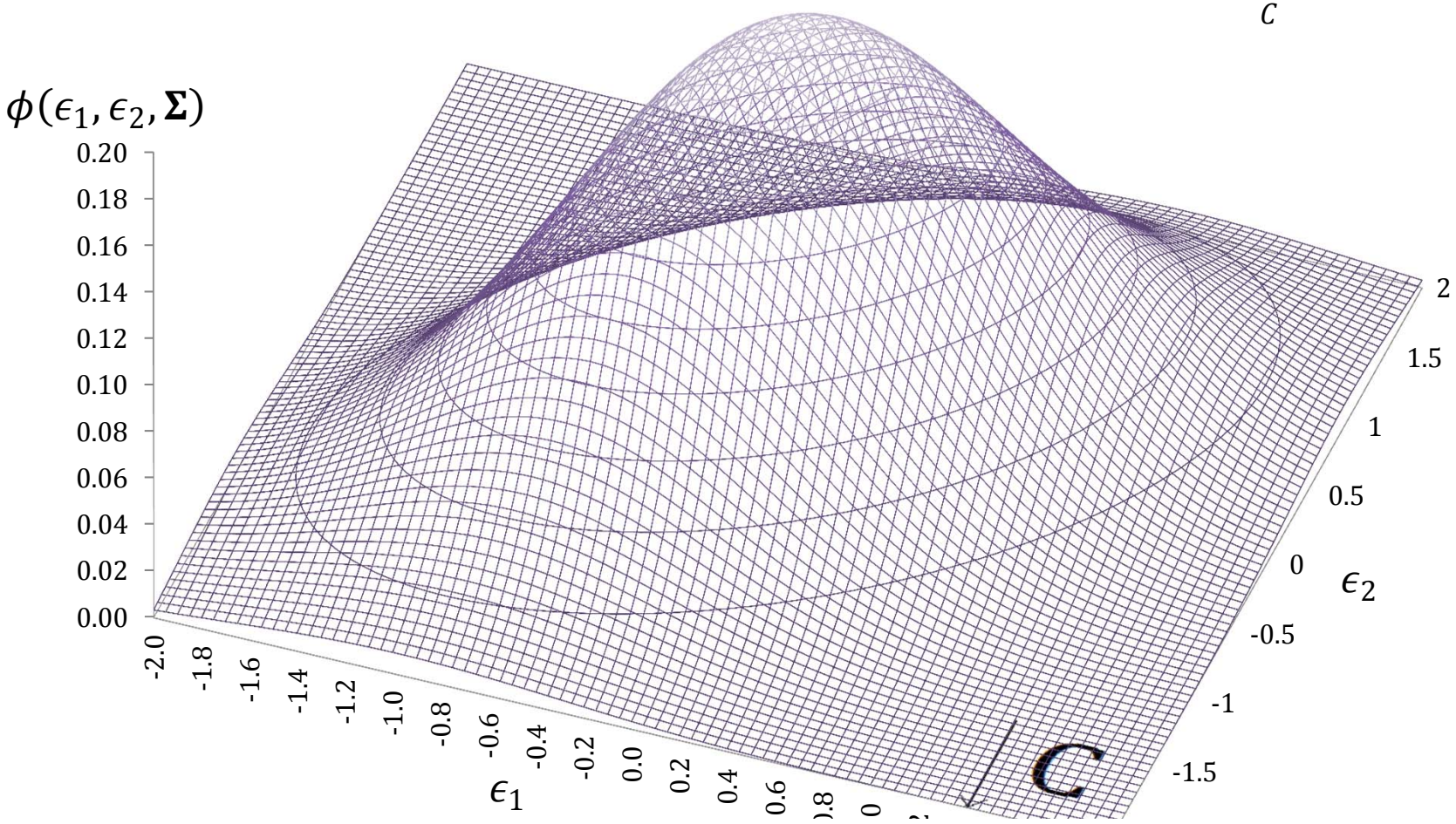
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (y_1, 0)'$

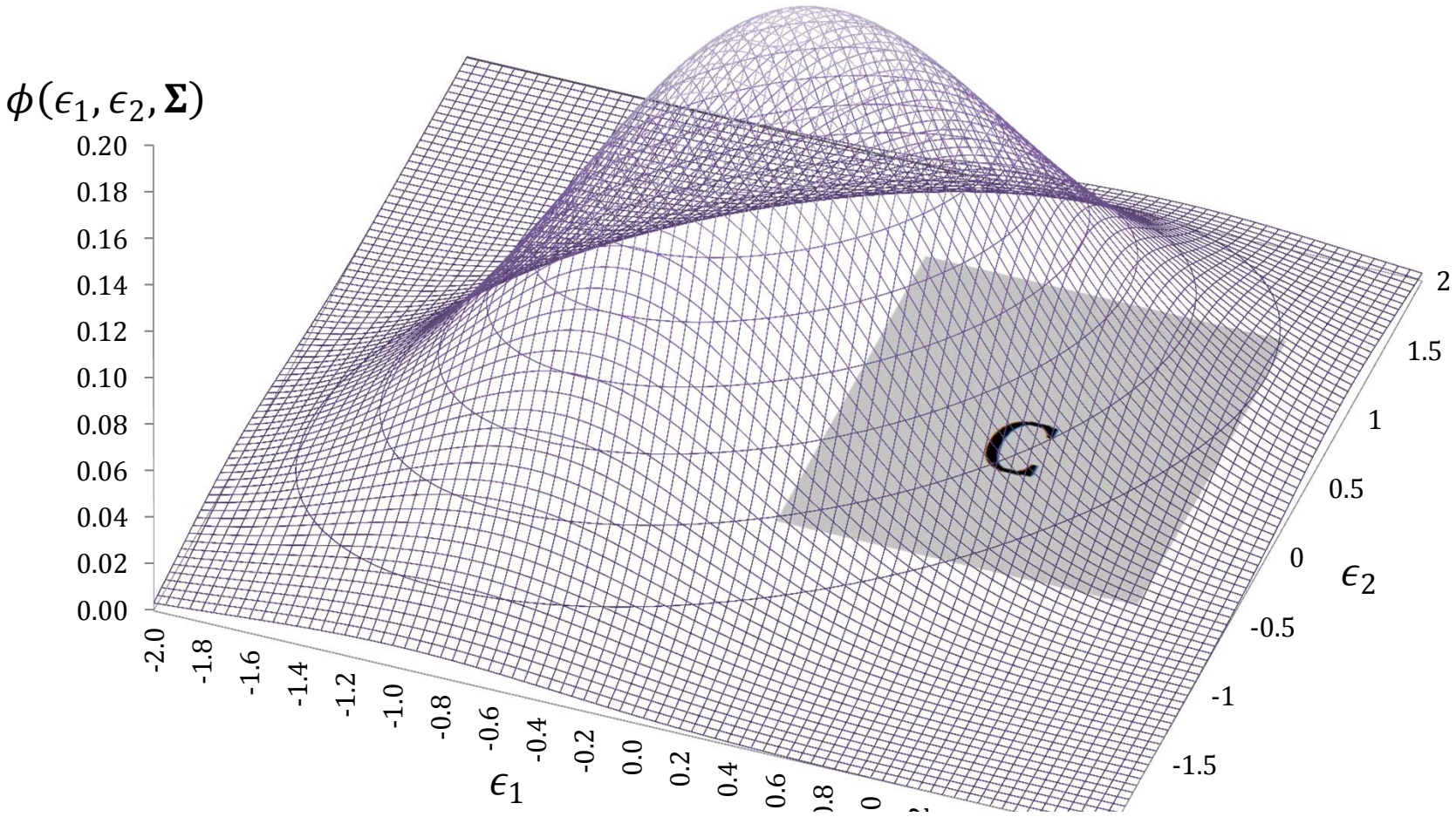
$\epsilon_1 = y_1 - \beta_1 x, \epsilon_2 < -\beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 = y_1 - \beta_1 x, \epsilon_2 < -\beta_2 x\}$

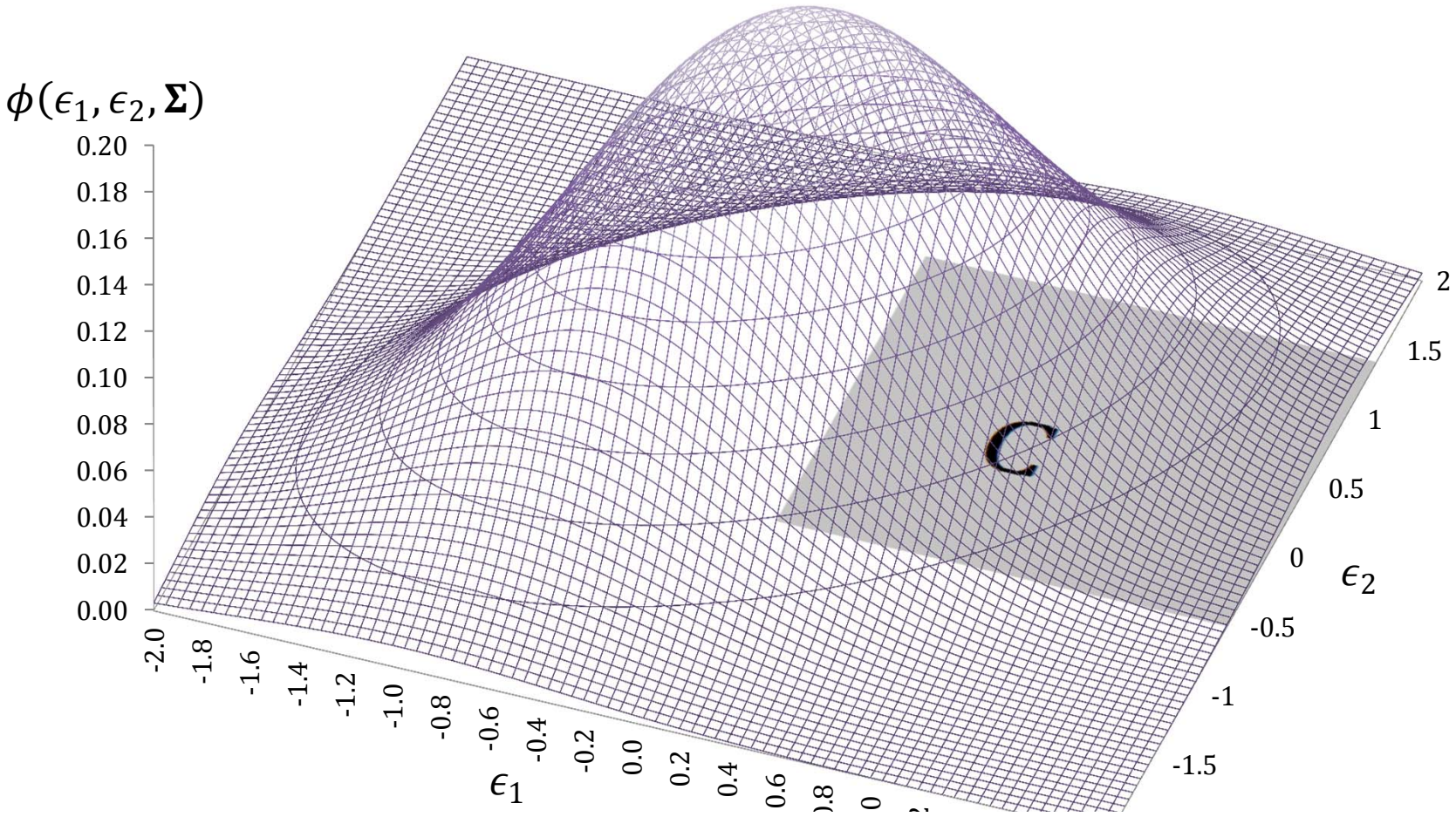
$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



Bivariate ordered probit



Mixed probit-ordered probit



$-\delta y_1$

Likelihood for SUR models

- Computed separately for each observation
  - So model can vary by observation
  - Different censoring structure, different # of eqs
- For each obs, integral of multivariate normal distribution
- Over some “Cartesian” region
  - Point, line, ray, half-plane, quarter-plane, etc.
- Core of `cmp` does this math

Two-stage probit

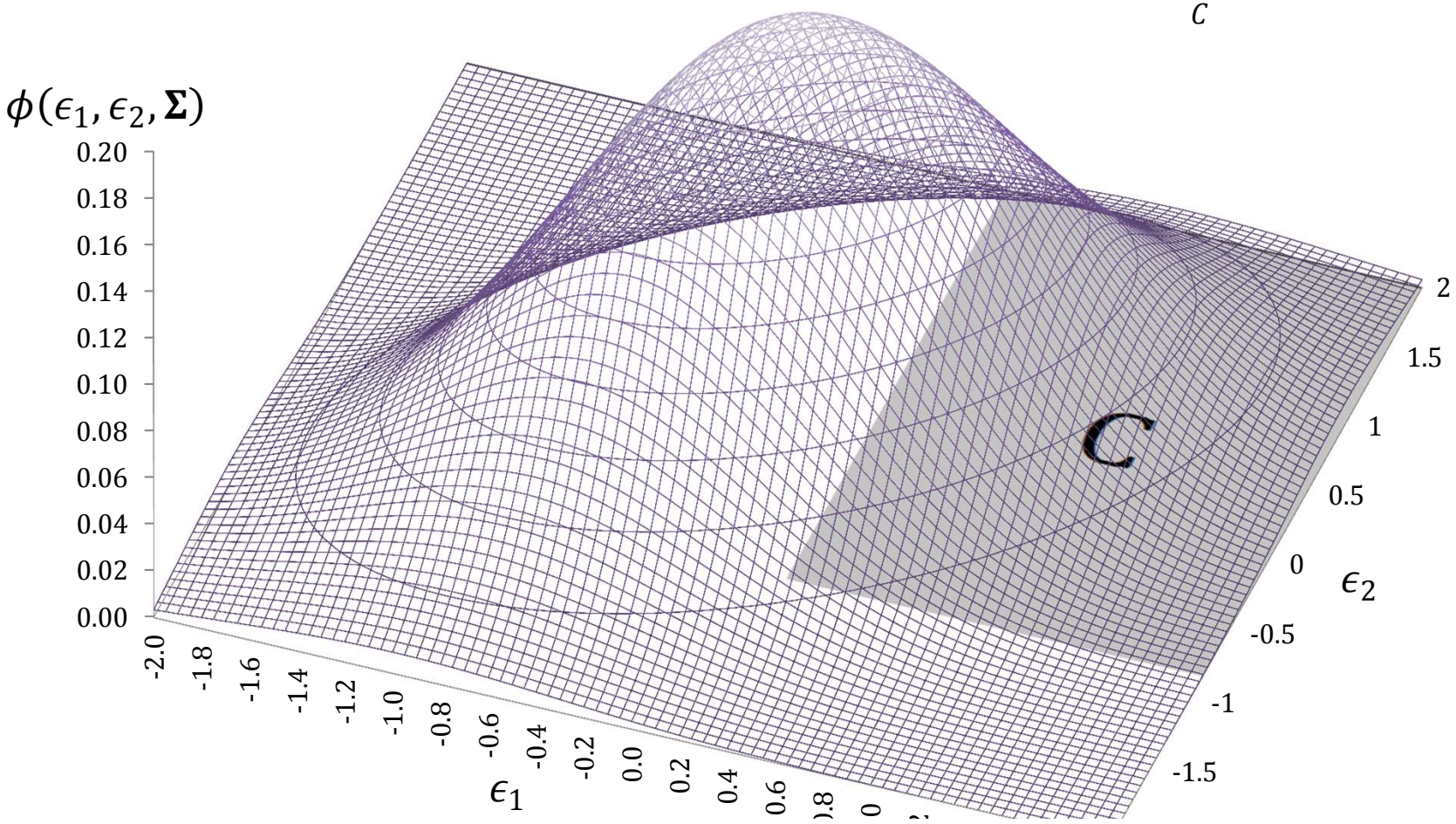
$$y_1^* = \beta_1 x + \epsilon_1$$
$$y_2^* = \delta y_1 + \beta_2 x + \epsilon_2$$
$$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$$
$$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$$

Case $\mathbf{y} = (1,1)'$

$$\epsilon_1 > -\beta_1 x, \epsilon_2 > -\delta y_1 - \beta_2 x$$

$$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 > -\beta_1 x, \epsilon_2 > -\delta y_1 - \beta_2 x\}$$

$$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$$



- Same math!
  - Still integrate normal distribution over Cartesian region
- `cmp` automatically multi-stage estimator
  - Can do IV, Heckman selection, switching regressions…
- By same token
  - `biprobit` can be a two-stage estimator
  - `ivregress` can do SUR
- Wasn't widely appreciated
  - Greene (1998): "surprisingly"…"seem not to be widely known"
  - Wooldridge (e-mail 2009): "I came to this realization somewhat late, although I've known it for a couple of years now."
- Requires system to be "fully observed" and recursive.

Available models

- Uncensored
- Tobit
- Probit
- Ordered probit
- Interval
- Multinomial probit
- Rank-ordered probit (new)

- Truncation now allowed in all but last two

- But wait! (three years)
- …there's more.
- No longer need:
  - fully observed
  - recursive

Two-stage probit with latent determinant

$y_1^* = \beta_1 x + \epsilon_1$

$y_2^* = \gamma y_1^* + \beta_2 x + \epsilon_2$

$\mathbf{y} = (\mathbf{1}\{y_1^* > 0\}, \mathbf{1}\{y_2^* > 0\})'$

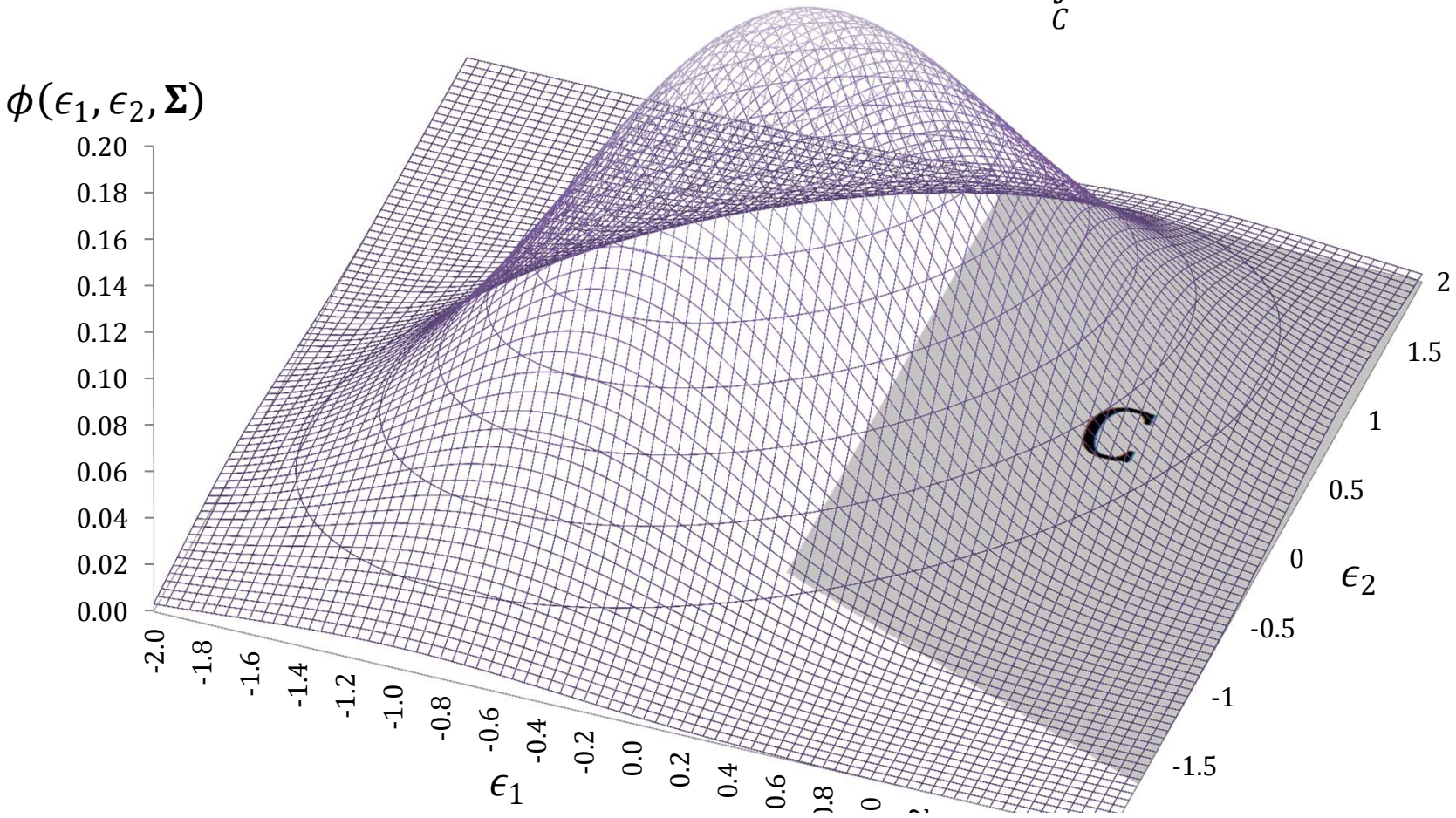
$\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)' \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma})$

Case $\mathbf{y} = (1,1)'$

$\epsilon_1 > -\beta_1 x, \epsilon_2 > -\gamma y_1^* - \beta_2 x = -\gamma\theta_1 - \gamma\epsilon_1 - \beta_2 x$

$C = \{(\epsilon_1, \epsilon_2)' : \epsilon_1 > -\beta_1 x, \epsilon_2 > -\gamma\theta_1 - \beta_2 x - \gamma\epsilon_1\}$

$\mathcal{L}(\beta_1, \beta_2, \boldsymbol{\Sigma}; \mathbf{y}_i | x_i) = \int_C \phi(\epsilon_1, \epsilon_2, \boldsymbol{\Sigma})$



I don’t know how to code that integral

Standard solution

$$
\begin{aligned}
y_1^* &= \beta_1 x + \epsilon_1 \\
y_2^* &= \gamma y_1^* + \beta_2 x + \epsilon_2 \\
&= \gamma(\beta_1 x + \epsilon_1) + \beta_2 x + \epsilon_2 \text{ (substituting)} \\
&= (\gamma\beta_1 + \beta_2)x + (\gamma\epsilon_1 + \epsilon_2) \text{ (rearranging)}
\end{aligned}
$$

Define:

$$
\begin{aligned}
\pi_1 &= \beta_1 \\
\pi_2 &= \gamma\beta_1 + \beta_2 \\
\boldsymbol{\omega} &= (\omega_1, \omega_2)' = (\epsilon_1, \gamma\epsilon_1 + \epsilon_2)'
\end{aligned}
$$

Then the system is:

$$
\begin{aligned}
y_1^* &= \pi_1 x + \omega_1 \\
y_2^* &= \pi_2 x + \omega_2 \\
\mathbf{y} &= (y_1^*, \mathbf{1}\{y_2^* > 0\})' \\
\boldsymbol{\omega} &\sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Omega})
\end{aligned}
$$

SUR system we know how to fit

Detail:

$$
\begin{aligned}
\boldsymbol{\Omega} = \mathbf{Cov}[\boldsymbol{\omega}] &= \begin{bmatrix} \mathrm{E}[\epsilon_1\epsilon_1] & \mathrm{E}[\epsilon_1(\gamma\epsilon_1 + \epsilon_2)] \\ \mathrm{E}[\epsilon_1(\gamma\epsilon_1 + \epsilon_2)] & \mathrm{E}[(\gamma\epsilon_1 + \epsilon_2)(\gamma\epsilon_1 + \epsilon_2)] \end{bmatrix} \\
&= \begin{bmatrix} 1 & \gamma + \rho \\ \gamma + \rho & \gamma^2 + 2\gamma\rho + 1 \end{bmatrix}
\end{aligned}
$$

More abstractly…

Rewrite: $$\begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix}' = \begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix}' \begin{bmatrix} 0 & \gamma \\ 0 & 0 \end{bmatrix} + x[\beta_1 \quad \beta_2] + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \end{bmatrix}'$$

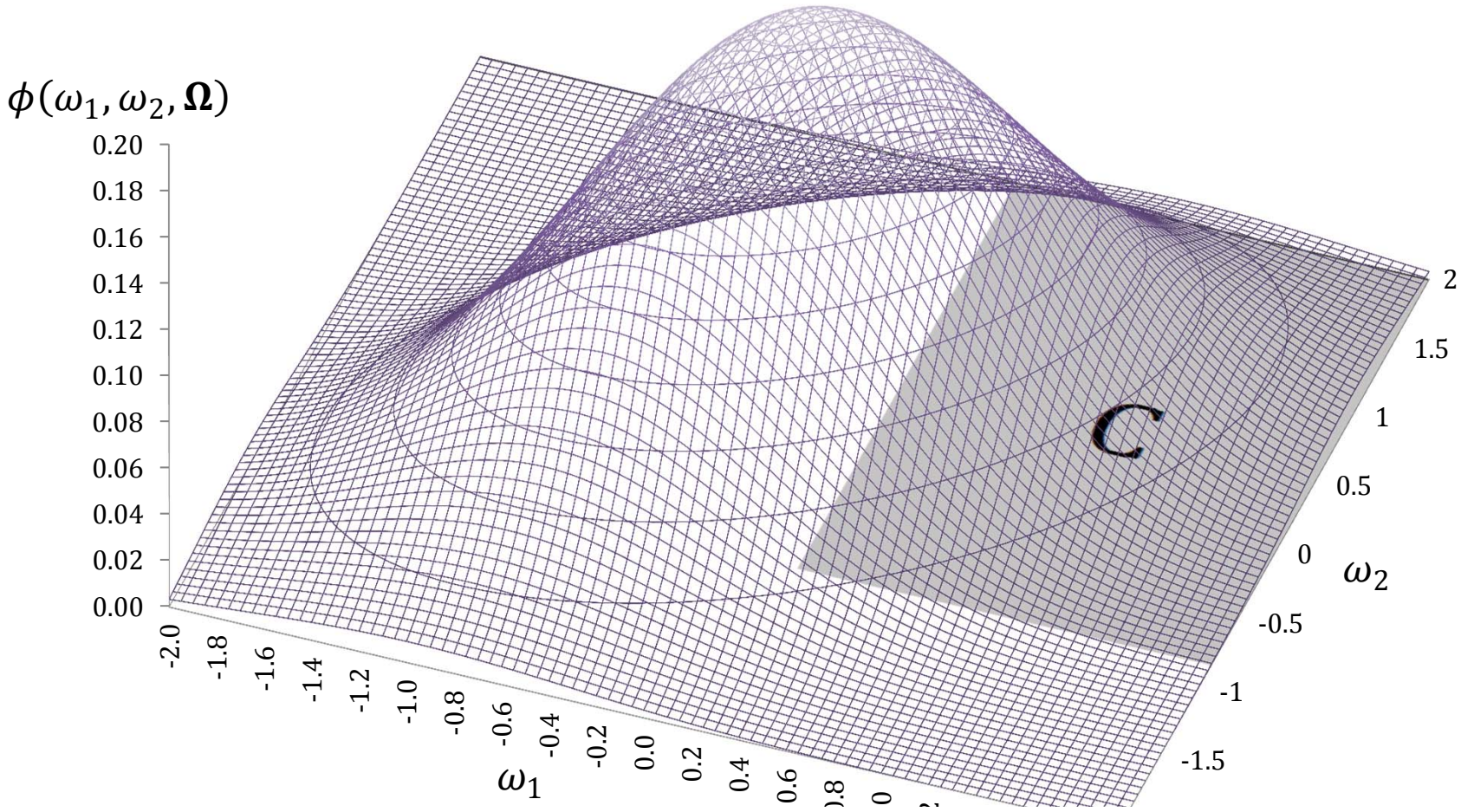
or: $$\mathbf{y}^{*\prime} = \mathbf{y}^{*\prime}\boldsymbol{\Gamma} + x\mathbf{B} + \boldsymbol{\epsilon}'$$

Solve: $$\begin{aligned} \mathbf{y}^{*\prime} &= x\mathbf{B}(\mathbf{I}-\boldsymbol{\Gamma})^{-1} + \boldsymbol{\epsilon}'(\mathbf{I}-\boldsymbol{\Gamma})^{-1} \\ &\equiv x\boldsymbol{\Pi} + \boldsymbol{\omega}' \end{aligned}$$

where: $$\boldsymbol{\Pi} = \mathbf{B}(\mathbf{I}-\boldsymbol{\Gamma})^{-1} \qquad \boldsymbol{\omega} = (\mathbf{I}-\boldsymbol{\Gamma})^{-1\prime}\,\boldsymbol{\epsilon}$$

and thus: $$\begin{aligned} \boldsymbol{\Omega} = \mathrm{E}[\boldsymbol{\omega}\boldsymbol{\omega}'] &= \mathrm{E}\left[(\mathbf{I}-\boldsymbol{\Gamma})^{-1\prime}\,\boldsymbol{\epsilon}\boldsymbol{\epsilon}'(\mathbf{I}-\boldsymbol{\Gamma})^{-1}\right] \\ &= (\mathbf{I}-\boldsymbol{\Gamma})^{-1\prime}\,\mathrm{E}[\,\boldsymbol{\epsilon}\boldsymbol{\epsilon}'](\mathbf{I}-\boldsymbol{\Gamma})^{-1} \\ &= (\mathbf{I}-\boldsymbol{\Gamma})^{-1\prime}\,\boldsymbol{\Sigma}(\mathbf{I}-\boldsymbol{\Gamma})^{-1} \end{aligned}$$

Multiplying system by $(\mathbf{I} - \mathbf{\Gamma})^{-1}$ transforms error space to make C Cartesian



```
sysuse auto, clear
cmp setup

* Fully observed
cmp (price = foreign ) (foreign = mpg), ind($cmp_cont $cmp_probit)

* With latent determinant
cmp (price = foreign#) (foreign = mpg), ind($cmp_cont $cmp_probit)
```

Math still works if Γ not triangular…

Rewrite:
$$\begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix}' = \begin{bmatrix} y_1^* \\ y_2^* \end{bmatrix}' \begin{bmatrix} 0 & \gamma_{12} \\ \gamma_{21} & 0 \end{bmatrix} + x[\beta_1 \quad \beta_2] + \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \end{bmatrix}'$$

or:
$$\mathbf{y}^{*'} = \mathbf{y}^{*'}\boldsymbol{\Gamma} + x\mathbf{B} + \boldsymbol{\epsilon}'$$

- References to linear functions need not be recursive
- Full simultaneity OK
- But references to censored variables must still be recursive

```
webuse klein, clear
cmp setup

* similar:

reg3 (consump    wagepriv  wagegovt) ///
     (wagepriv   consump   govt capital1)

cmp  (consump  = wagepriv# wagegovt) ///
     (wagepriv = consump#  govt capital1), ///
     ind($cmp_cont $cmp_cont) nolr tech(dfp) qui
```

Random effects and coefficients

- Hierarchical/multilevel
- At each level, can be correlated within and across equations
  - (or not)
- Handled using simulation or (adaptive) quadrature
  - Adaptive quadrature not available yet with more than two levels

```
webuse union, clear

* Random efect:
xtprobit union age grade c.south##c.year
cmp   (union = age grade c.south##c.year || idcode:), ///
        ind($cmp_probit) tech(dfp) nolr

* Random coefficient too, potentially correlated with random effect:
cmp   (union = age grade c.south##c.year || idcode: grade), ///
        ind($cmp_probit) tech(dfp) nolr

* Random coefficient too, independent of random effect:
cmp   (union = age grade c.south##c.year || idcode: grade, cov(ind)), ///
        ind($cmp_probit) tech(dfp) nolr intpoints(8)

* Random coefficient only:
cmp   (union = age grade c.south##c.year || idcode: grade, nocons), ///
        ind($cmp_probit) tech(dfp) nolr intpoints(8)
```

Probit with random effect

$y_{it}^* = \beta x_{it} + \nu_i + \epsilon_{it}$
$y_{it} = \mathbf{1}\{y_{it}^* > 0\}$
$\nu_i \sim \mathcal{N}(0, \sigma_\nu^2)$
$\epsilon_{it} | \nu_i \sim \mathcal{N}(0, \sigma^2)$

To compute likelihood, need to integrate probability density for ν_i, ϵ_{it} over all values of *both* that are together compatible with the y_{it} we observe.

For each group i, integrate over possible values of random effect—weighting by probability of effect of that value—of product of resulting likelihoods of the observations in the group:

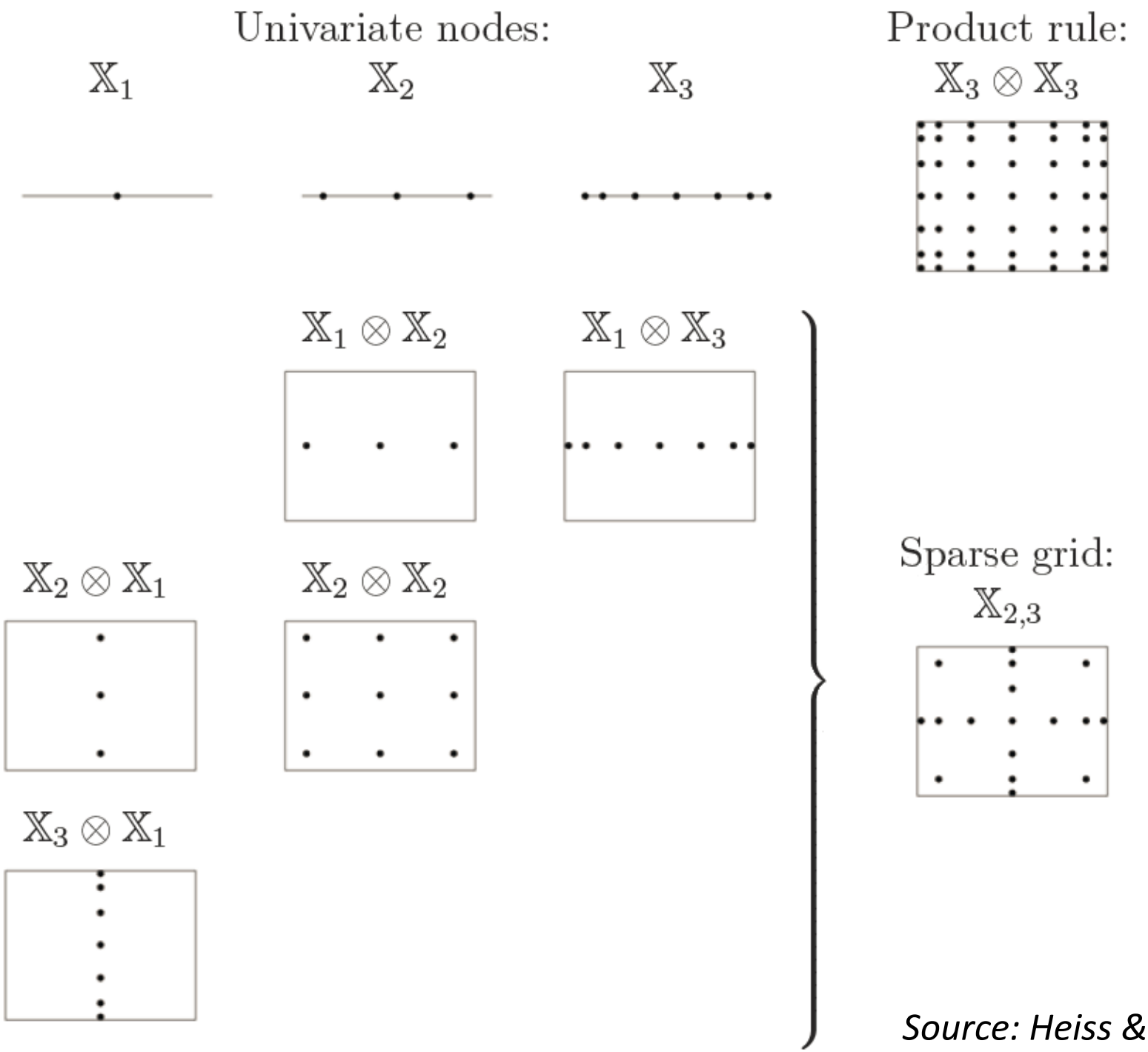
$$\mathcal{L}(\beta, \sigma_\nu^2, \sigma^2; \mathbf{y}_i | \mathbf{x}_i) = \int_{-\infty}^{\infty} \phi(\nu_i, \sigma_\nu^2) \left[\prod_t \mathcal{L}(\beta, \sigma^2; y_{it} | x_{it}, \nu_i) \right] d\nu_i$$

Inner likelihood computed as in non–random effects models:

$$\mathcal{L}(\beta, \sigma^2; \mathbf{y}_i | \mathbf{x}_i, \nu_i) = \int_C \phi(\epsilon_{it}, \sigma^2)$$

Numerical method I: Quadrature

- Evaluate function at ~4–16 special points with special weights
- Traditional strength: computationally cheap
- Weak if integrated function is a narrow bump between special points; Rabe-Hesketh, Skrondal, Pickles 2002)
- Adaptive quadrature fixes by searching for the bump; slower
- Evaluations increase exponentially in dimensions of integration
  - Integrating over 3-D effect can take 4^3–16^3 evaluations
  - BUT “sparse grids” can help (Heiss & Winschel 2008)



Source: Heiss & Winschel 2008

Numerical method I: Simulation

- Sample integrated function at 50, 100, 500 points…
- Far more CPU time than quadrature
  - Maybe not more than adaptive quadrature?
- Extends naturally to higher dimensions
  - Multidimensional Halton sequences (Cappellari & Jenkins 2006, Drukker & Gates 2006)
  - Can “scramble” (Kolenikov 2012)
  - Can “step”: first estimate coarsely with 10 points, then refine with 20, 40…

`cmp` offers (adaptive) quadrature and simulation

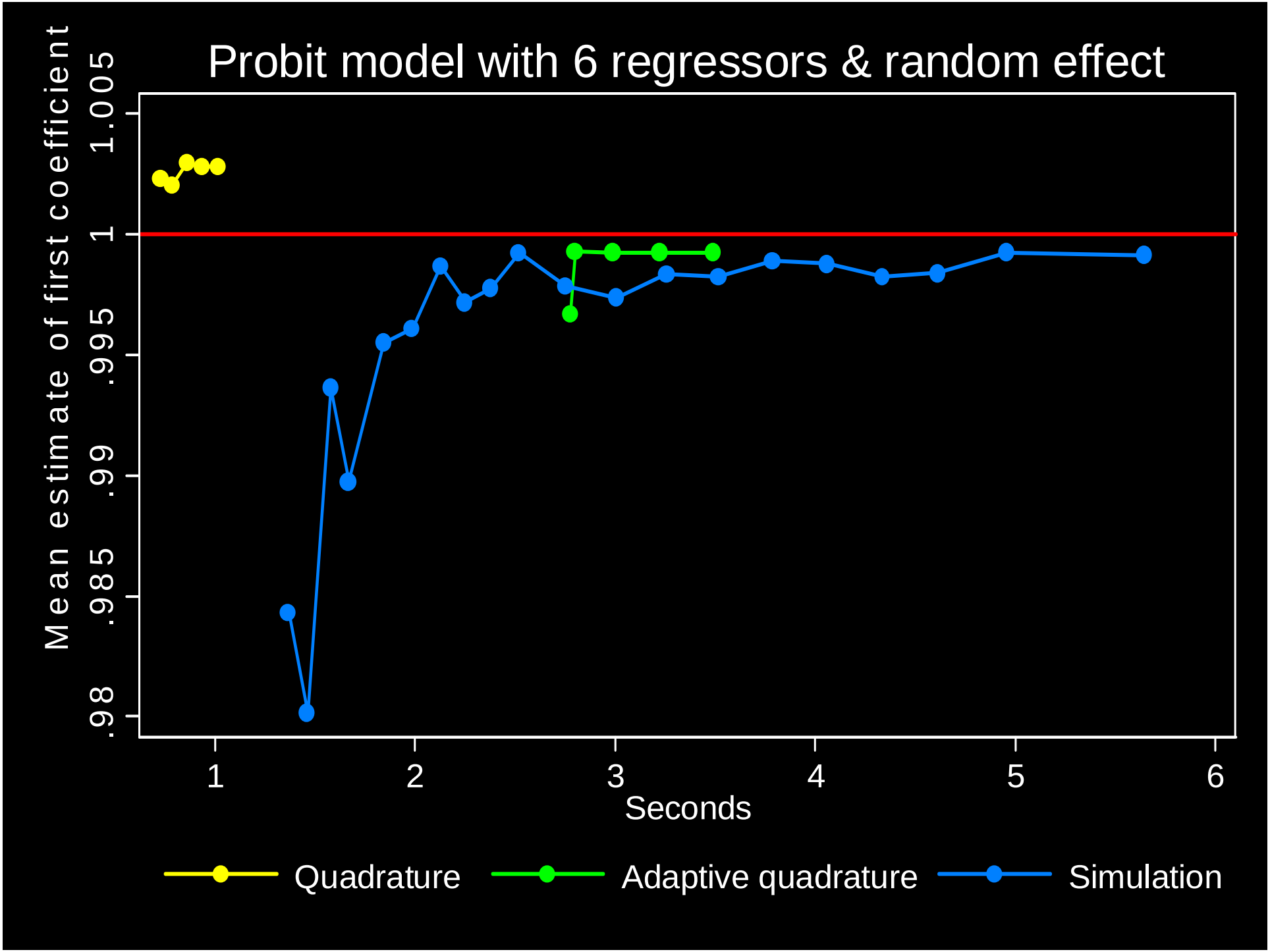
- But adaptive quadrature not ready for models with >2 levels

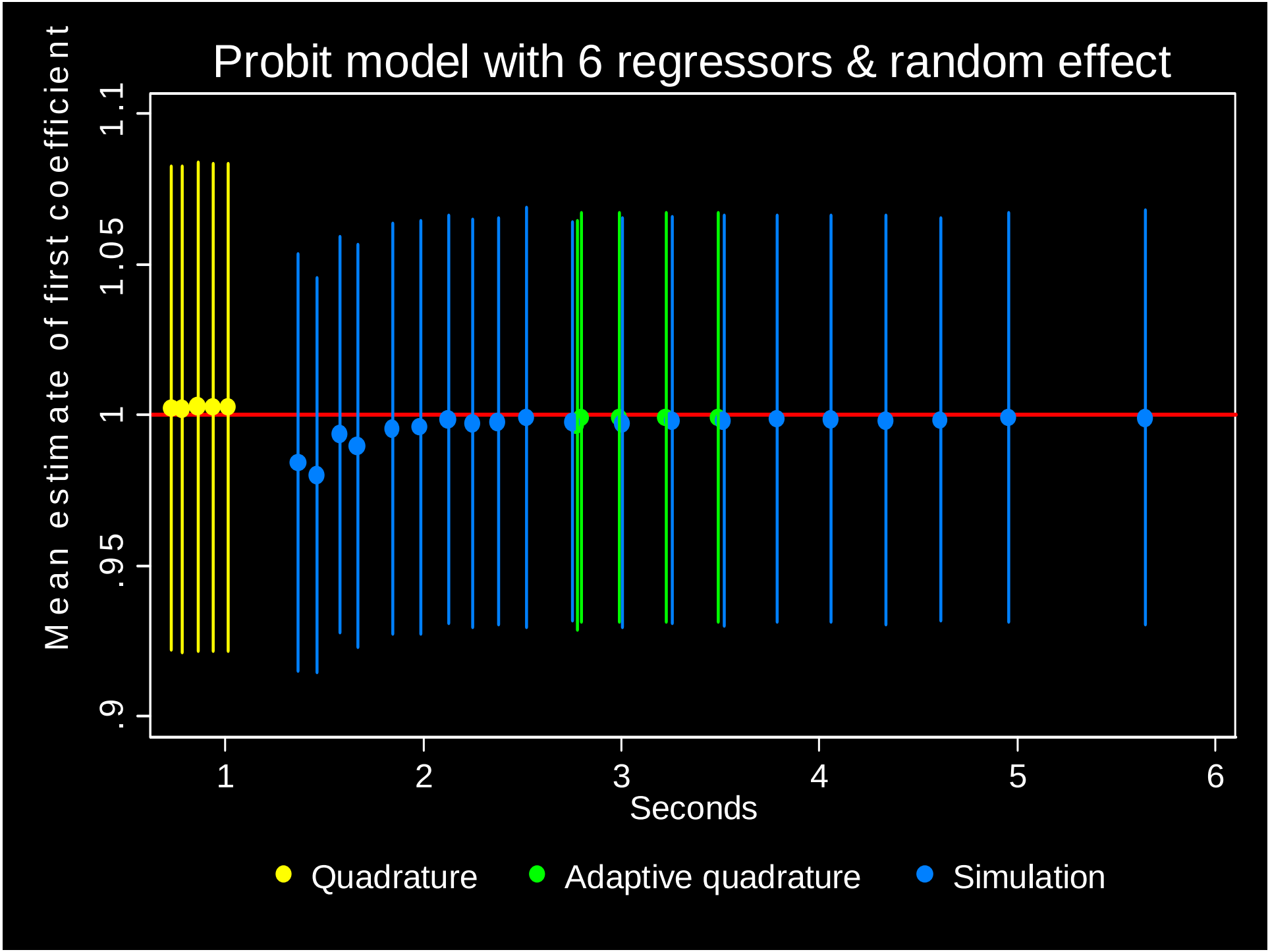
Computation trick: pseudo-linear form

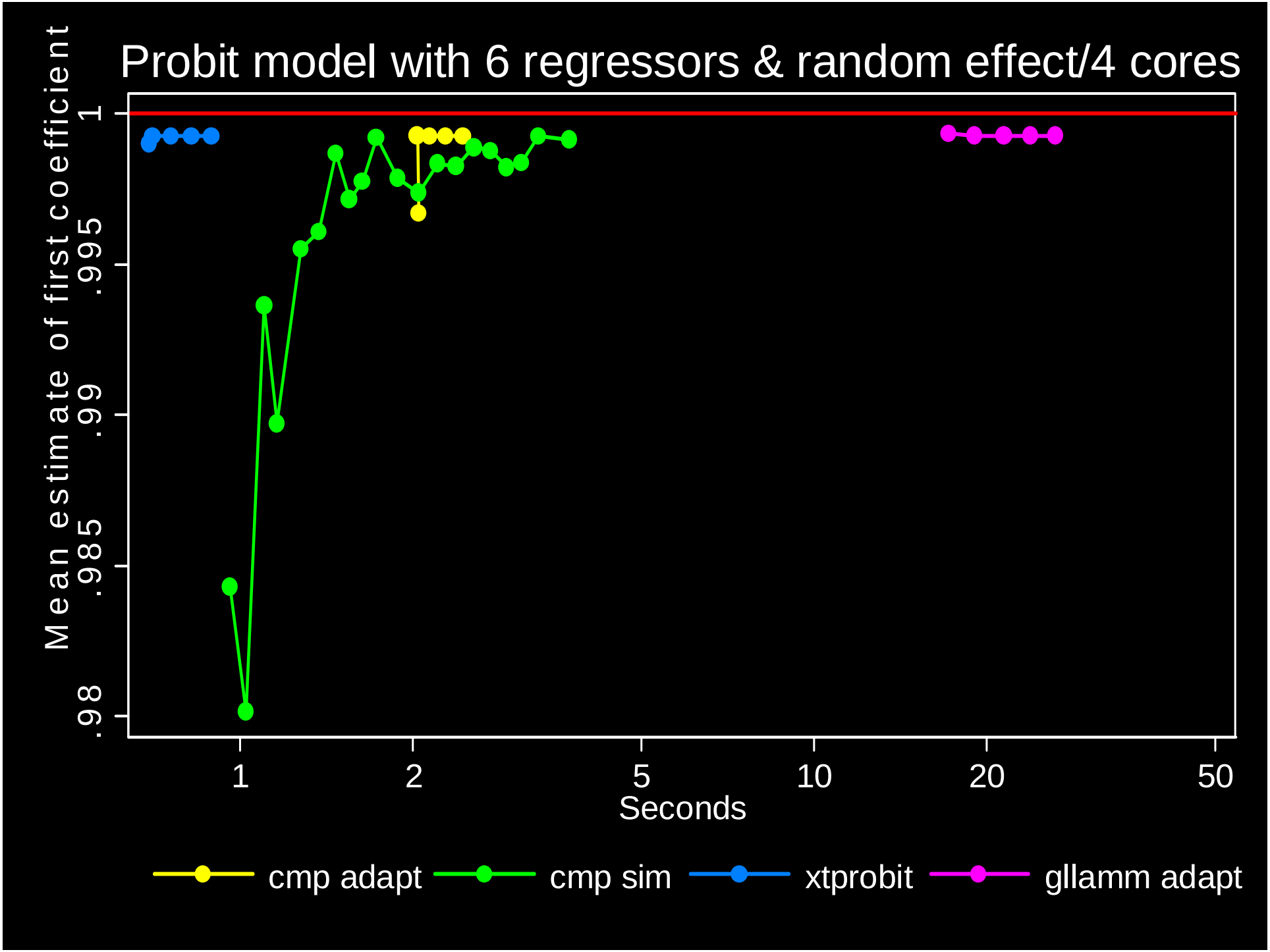
- In Stata ML, new "lf1" evaluator type meets my plea
- Computationally efficient when
  - Evaluator program computes 1st derivates analytically, but not 2nd
  - Model has linear components $\theta_j = \mathbf{x}'\boldsymbol{\beta}_j$
    - Evaluator does $\frac{\partial \ln \mathcal{L}}{\partial \theta_j}$. Stata does $\frac{\partial \ln \mathcal{L}}{\partial \boldsymbol{\beta}_j} = \mathbf{x}' \frac{\partial \ln \mathcal{L}}{\partial \theta_j}$.
    - Stata computes 2nd derivatives numerically: just one evaluator call per θ_j, not per element of $\boldsymbol{\beta}_j$.
- Requires log likelihood (LL) to have "linear form"
  - Full LL is sum of LLs for each observation
  - Not true in random effects models: LL is a trait of groups
- Trick. For each observation, evaluator provides:
  - Fake LL ≡ full LL / # of observations
  - Real $\frac{\partial \ln \mathcal{L}}{\partial \theta_j}$
  - Correctly, Stata still does $\frac{\partial \ln \mathcal{L}}{\partial \boldsymbol{\beta}_j} = \mathbf{x}' \frac{\partial \ln \mathcal{L}}{\partial \theta_j}$

Computing time for Hessian

- `gllamm`
  - quadratic in # of parameters
- `cmp`
  - linear in # of linear components + auxiliary parameters


Probit model with 6 regressors & random effect
Mean estimate of first coefficient
.98
.985
.99
.995
1
1.005
1
2
3
4
5
6
Seconds
Quadrature
Adaptive quadrature
Simulation


Probit model with 6 regressors & random effect
Mean estimate of first coefficient
.9
.95
1
1.05
1.1
1
2
3
4
5
6
Seconds
Quadrature
Adaptive quadrature
Simulation


Probit model with 6 regressors & random effect/4 cores
Mean estimate of first coefficient
1
.995
.99
.985
.98
1
2
5
10
20
50
Seconds
cmp adapt
cmp sim
xtprobit
gllamm adapt