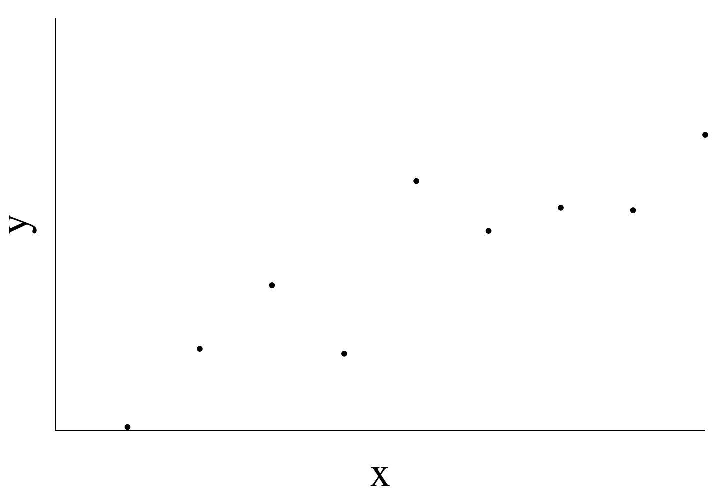
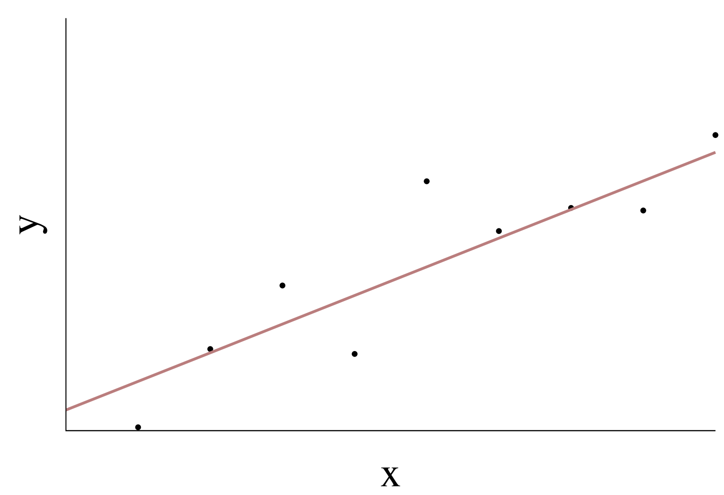
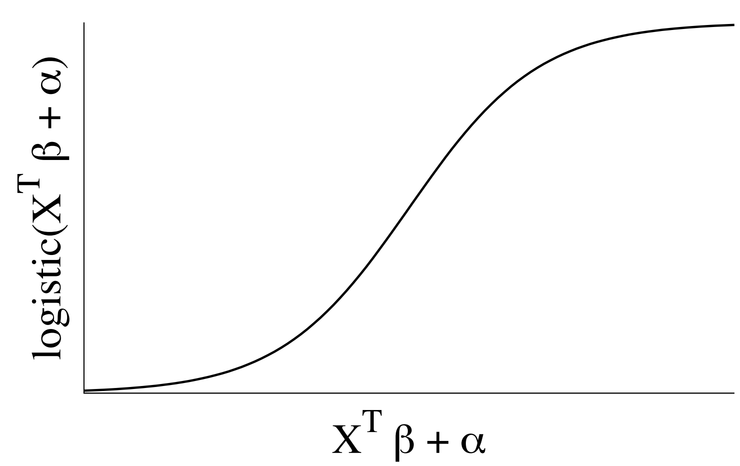
Linear and Generalized Linear Models







We still just need to specify posterior density

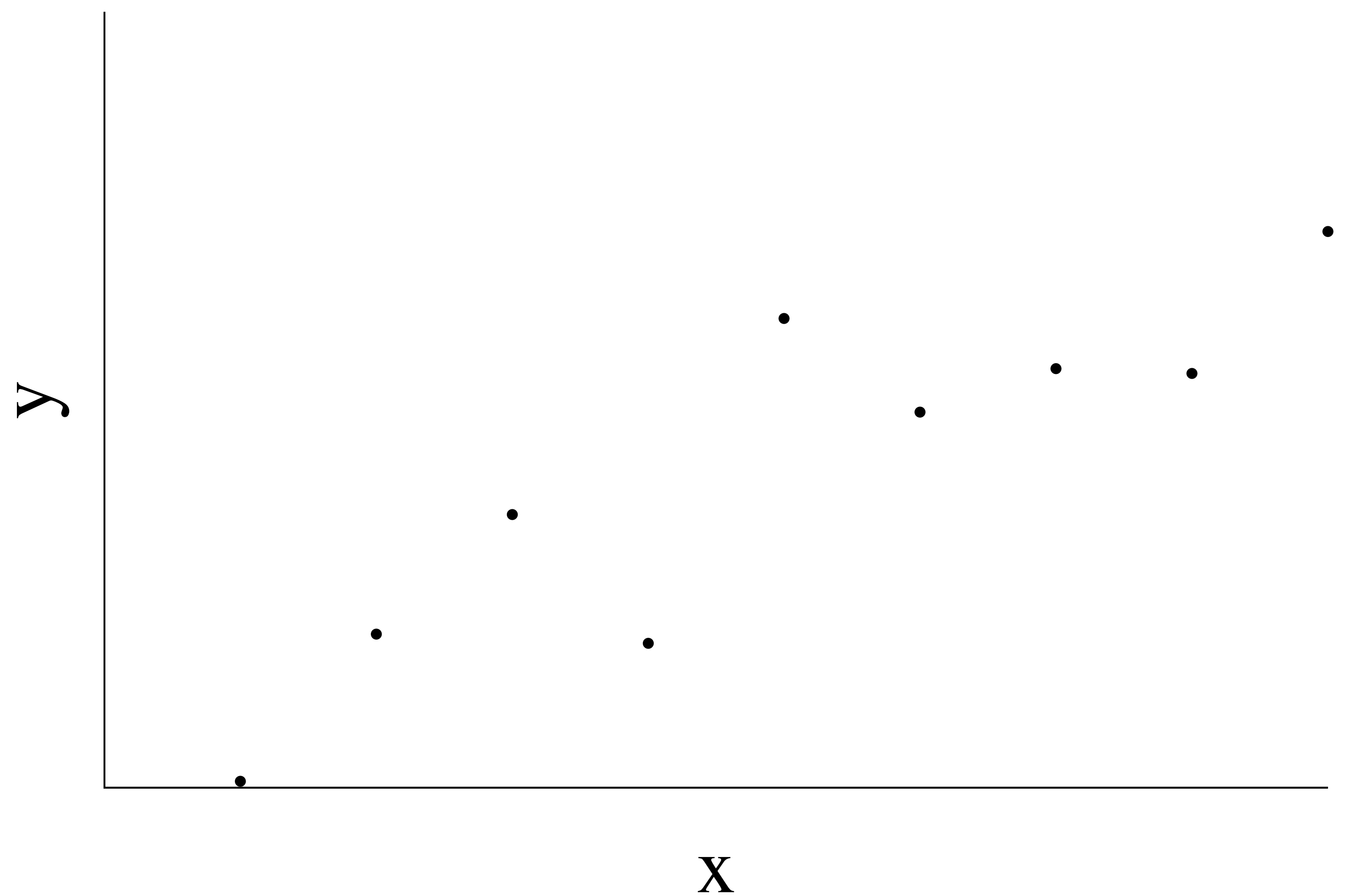
$$p(q \mid \mathcal{D})$$

Often the data naturally separates into *variates*, y, and *covariates*, x.

(we'll say *outcomes*, y, and *predictors*, x)

$$\mathcal{D} \to \{y, x\}$$

Regression models the statistical relationship between the outcome and the predictor(s).



Regression models the statistical relationship between the outcome and the predictor(s).

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

Regression models the statistical relationship between the outcome and the predictor(s).

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

We typically assume that the predictors (covariates) are independent of the model parameters.

$$p(x|\theta) = p(x)$$

In which case the likelihood becomes a model of the outcome *conditional on* the predictors.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

In which case the likelihood becomes a model of the outcome *conditional on* the predictors.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

$$p(y, x|\theta) = p(y|x, \theta)\, p(x)$$

In which case the likelihood becomes a model of the outcome *conditional on* the predictors.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

$$p(y, x|\theta) = p(y|x, \theta)\, p(x)$$

$$p(y, x|\theta) \propto p(y|x, \theta)$$

In which case the likelihood becomes a model of the outcome *conditional on* the predictors.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

Independence assumption

$$p(y, x|\theta) = p(y|x, \theta)\, p(x)$$

$$p(y, x|\theta) \propto p(y|x, \theta)$$

We'll make this assumption for now,
but it's not always valid.

$$p(y, x|\theta) = p(y|x, \theta)\, \boxed{p(x|\theta)}$$

Independence
assumption

$$p(y, x|\theta) = p(y|x, \theta)\, \boxed{p(x)}$$

We'll make this assumption for now,
but it's not always valid.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

Selection
Bias

$$p(y, x|\theta) = p(y|x, \theta)\, p(x)$$

We'll make this assumption for now, but it's not always valid.

$$p(y, x|\theta) = p(y|x, \theta)\, p(x|\theta)$$

Selection Bias

$$p(y, x|\theta) \cancel{=} p(y|x, \theta)\, p(x)$$

Predictors are often restricted to a single effective parameter through a deterministic mapping.

$$p(y|x,\theta) = p(y|f(x,\theta_1),\theta_2)$$

Predictors are often restricted to a single effective parameter through a deterministic mapping.

$$p(y|x,\theta) = p(y|f(x,\theta_1),\theta_2)$$

Predictors are often restricted to a single effective parameter through a deterministic mapping.

$$p(y|x,\theta) = p(y|f(x,\theta_1),\theta_2)$$

$$p(y|x,\theta) = \mathcal{N}(y|f(x,\theta),\sigma)$$

Predictors are often restricted to a single effective parameter through a deterministic mapping.

$$p(y|x,\theta) = p(y|f(x,\theta_1),\theta_2)$$

$$p(y|x,\theta) = \mathcal{N}(y|f(x,\theta),\sigma)$$

$$p(y|x,\theta) = \mathrm{Bin}(y|f(x,\theta),N)$$

This immediately generalizes to multiple effective parameters.

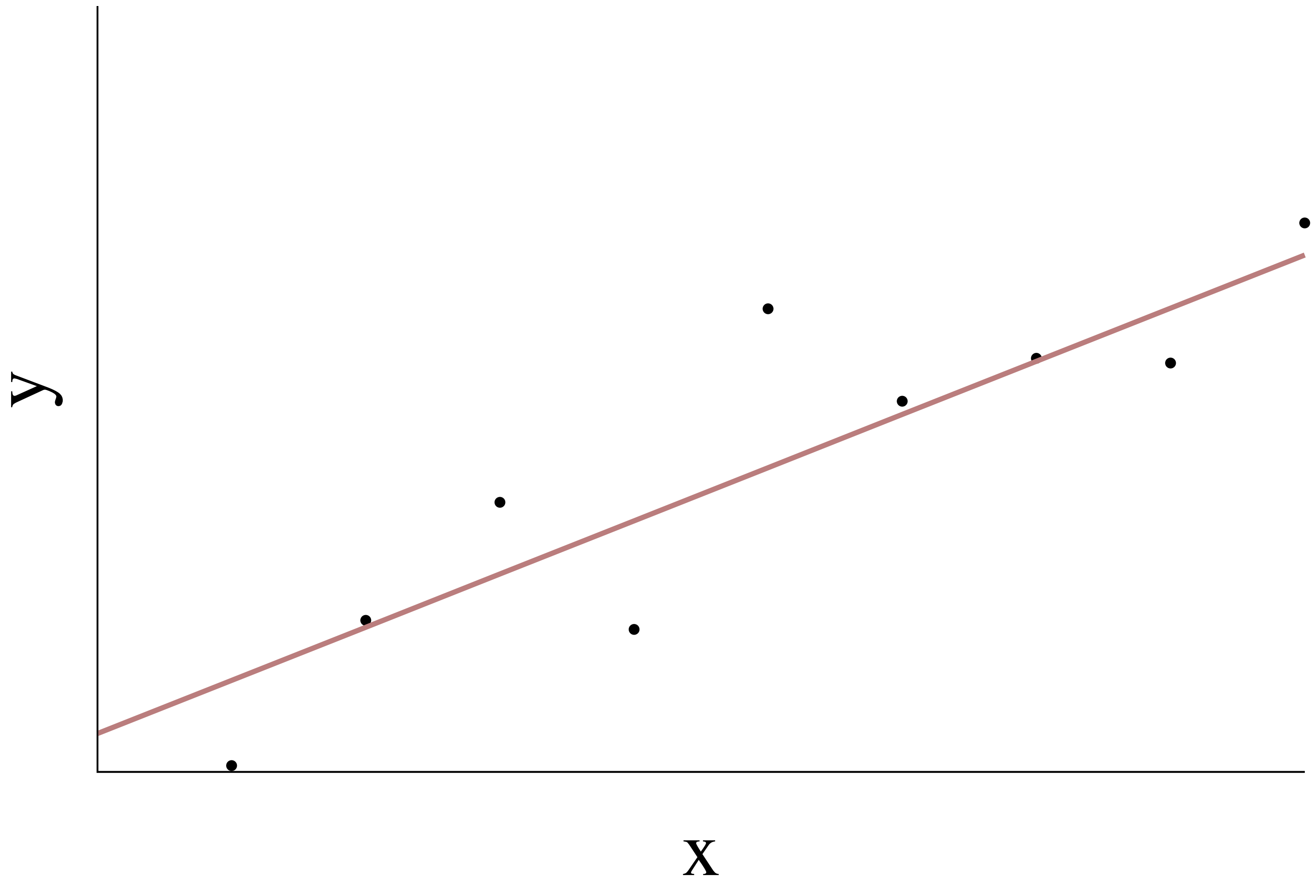
$$p(y|x, \theta) = p(y|f_1(x, \theta_1), f_2(x, \theta_2), \theta_3)$$

This immediately generalizes to multiple effective parameters.

$$p(y|x,\theta) = p(y|f_1(x,\theta_1), f_2(x,\theta_2), \theta_3)$$

$$p(y|x,\theta) = \mathrm{Gamma}(y|\alpha(x,\theta_1), \beta(x,\theta_2))$$

Linear Models



When an effective parameter is unconstrained we can model it with a linear mapping.

$$f(x, \alpha, \beta) = \beta \cdot x + \alpha$$

Multiple covariates are commonly encapsulated in a *design matrix*.

$$f(x, \alpha, \boldsymbol{\beta}) = \sum_{n,i} X_{in} \beta_i + \alpha$$

Multiple covariates are commonly encapsulated in a *design matrix*.

$$f(x, \alpha, \boldsymbol{\beta}) = \sum_{n,i} X_{in} \beta_i + \alpha$$

$$f(x, \alpha, \boldsymbol{\beta}) = \mathbf{X}^T \boldsymbol{\beta} + \alpha$$

Multiple covariates are commonly encapsulated in a *design matrix*.

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[K, N] X;
}
```

Multiple covariates are commonly encapsulated in a *design matrix*.

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[K, N] X;
}
parameters {
  vector[K] beta;
  real alpha;
  ...
}
```

Multiple covariates are commonly encapsulated in a *design matrix*.

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[K, N] X;
}
parameters {
  vector[K] beta;
  real alpha;
  ...
}
model {
  vector[N] y_tilde;
  y_tilde = X’ * beta + alpha;
  ...
}
```

Avoid transposing X every time

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[K, N] X;
}
parameters {
  vector[K] beta;
  real alpha;
  ...
}
model {
  vector[N] y_tilde;
  y_tilde = X’ * beta + alpha;
  ...
}
```

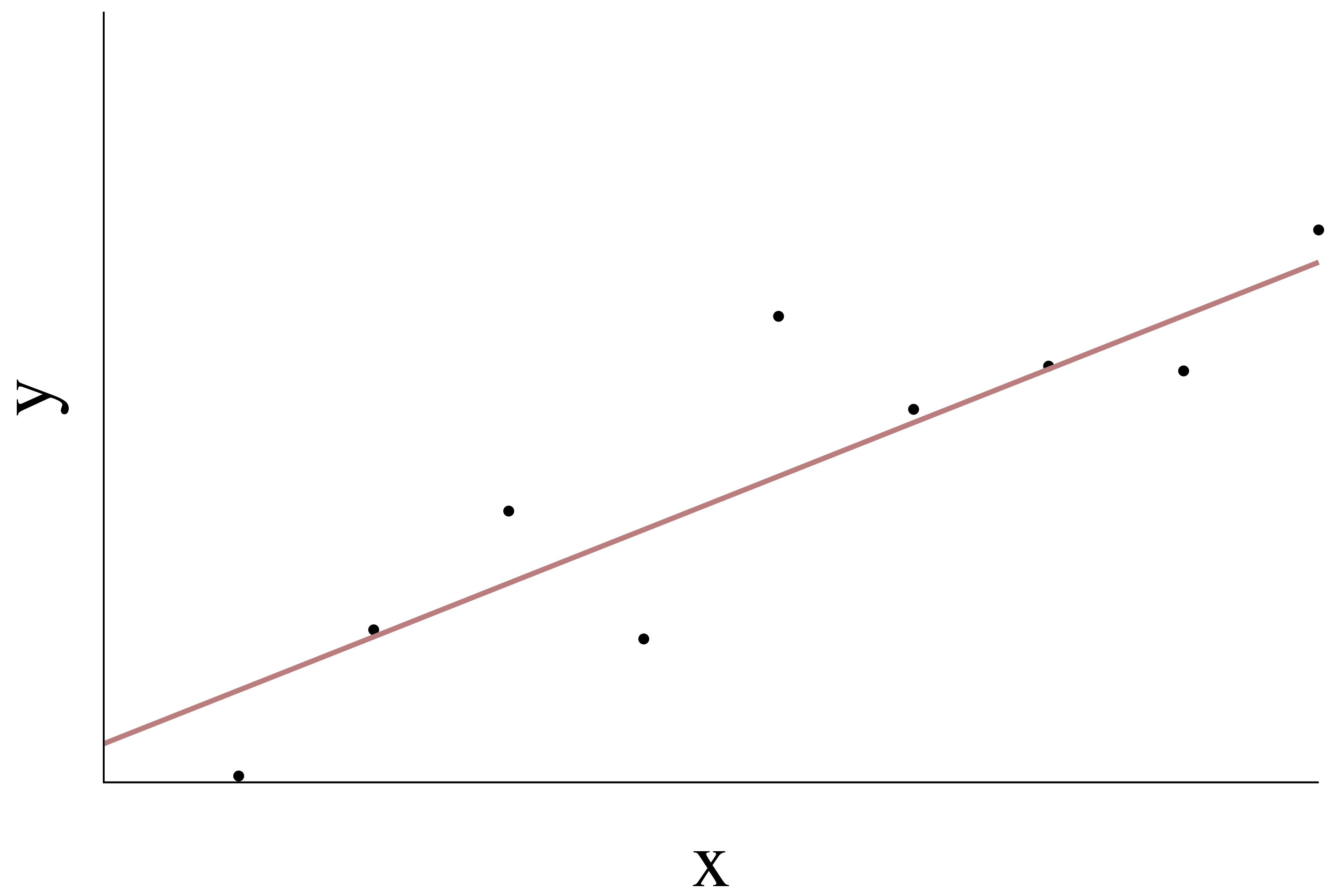
Avoid transposing X every time

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  ...
}
model {
  vector[N] y_tilde;
  y_tilde = X' * beta + alpha;
  ...
}
```

Avoid transposing X every time

```
data {
  int N; // Sample size
  int K; // Number of predictors
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  ...
}
model {
  vector[N] y_tilde;
  y_tilde = X * beta + alpha;
  ...
}
```

When the measurement model is Gaussian we recover the ubiquitous Gaussian-Linear model.



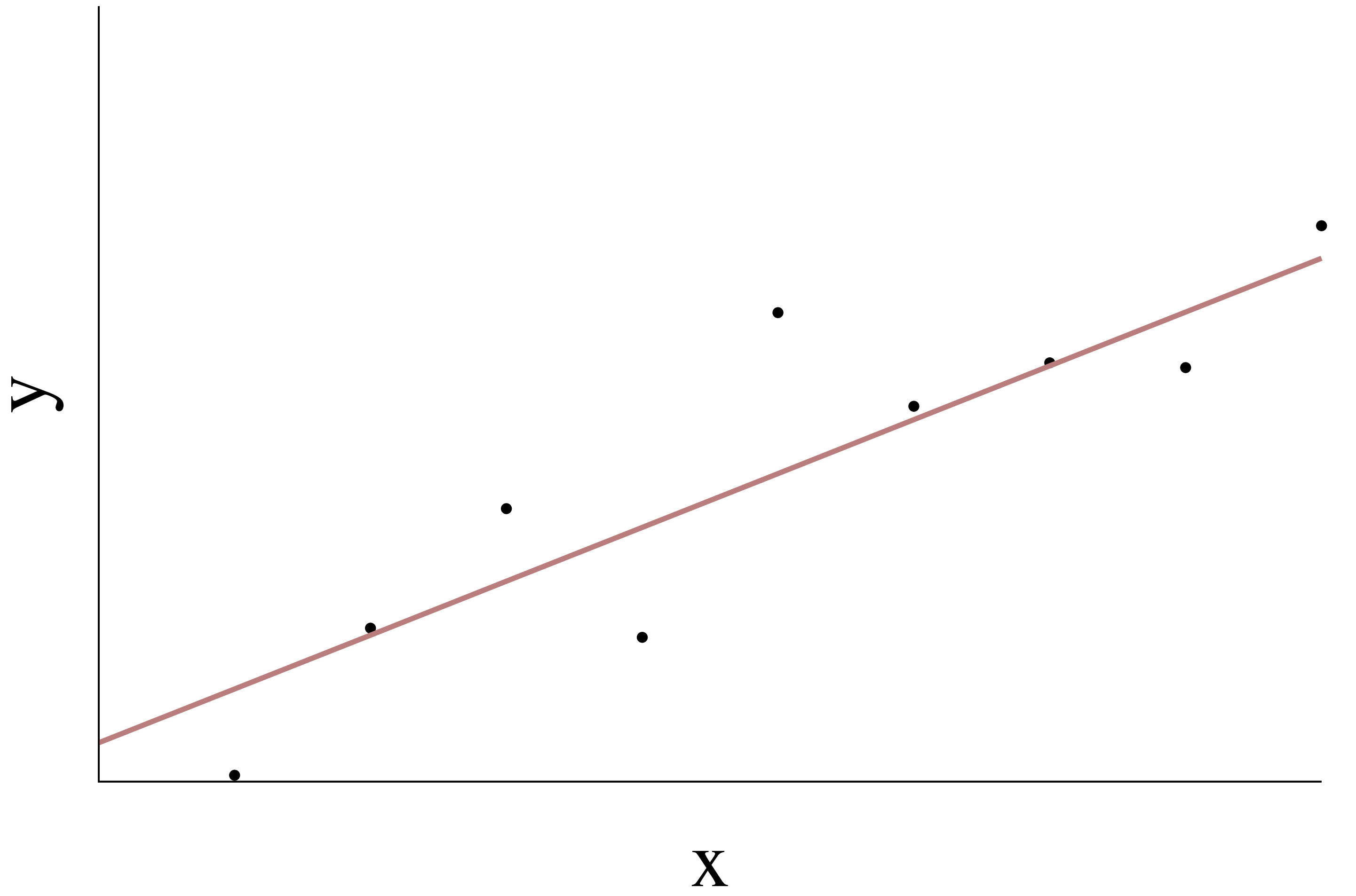
$$p(y|\mathbf{X}, \alpha, \boldsymbol{\beta}, \sigma) = \mathcal{N}\left(y|\mathbf{X}^T\boldsymbol{\beta} + \alpha, \sigma\right)$$

When the measurement model is Gaussian we recover the ubiquitous Gaussian-Linear model.

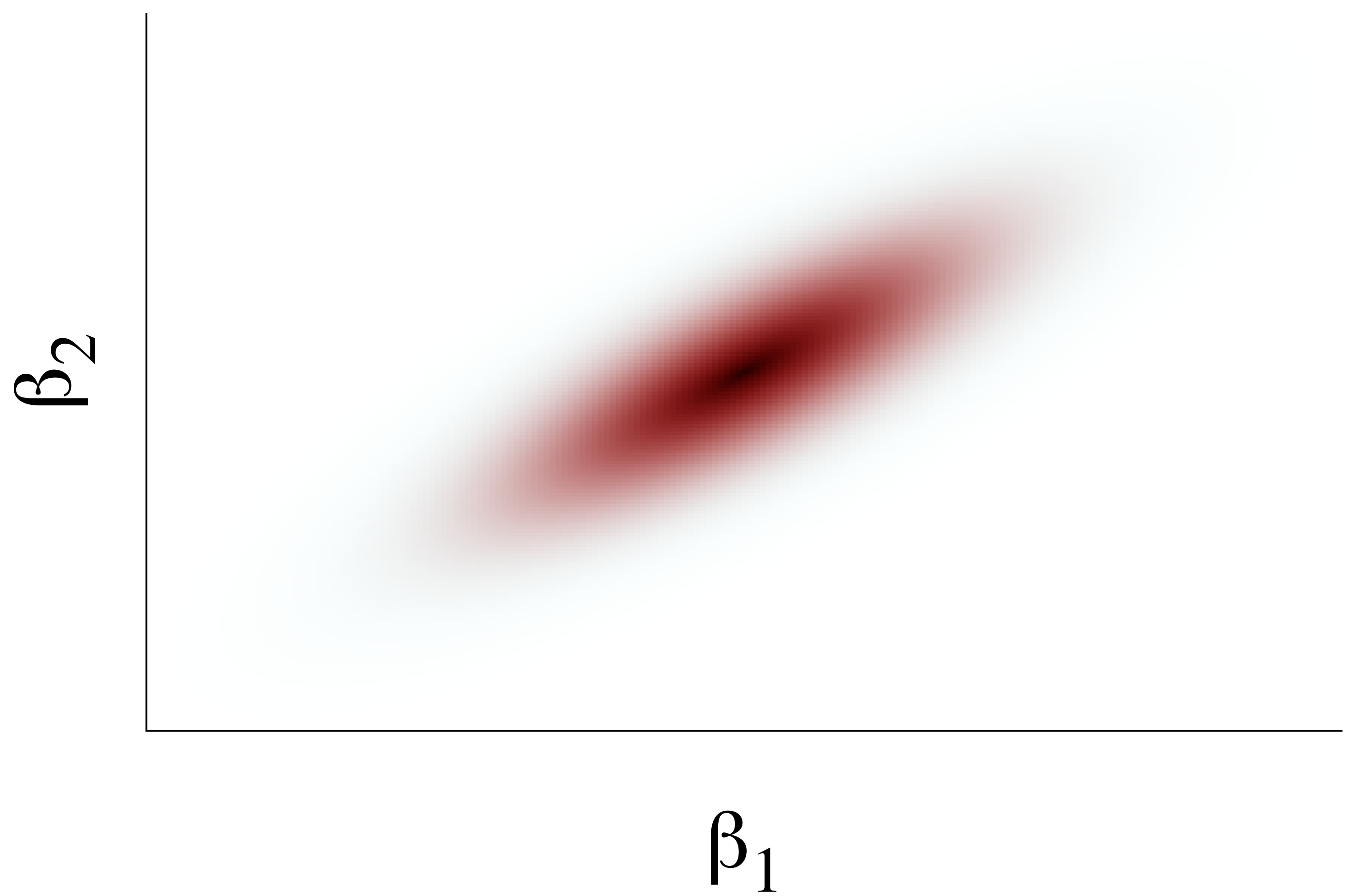
```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);

  // prior for beta?
  // prior for alpha?
  // prior for sigma?
}
```

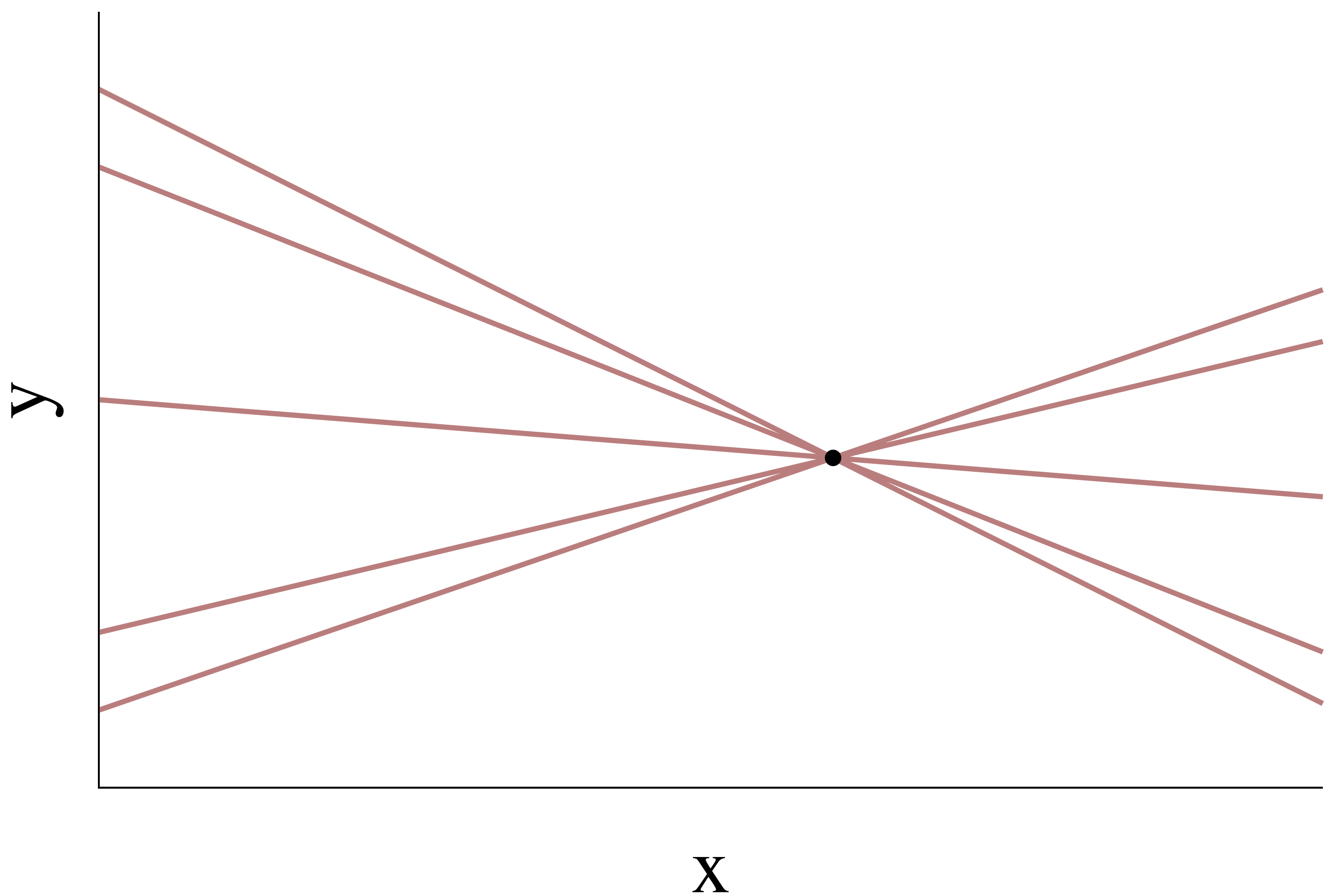
Given enough data, linear models are over-constrained and all the slopes can be fit well.



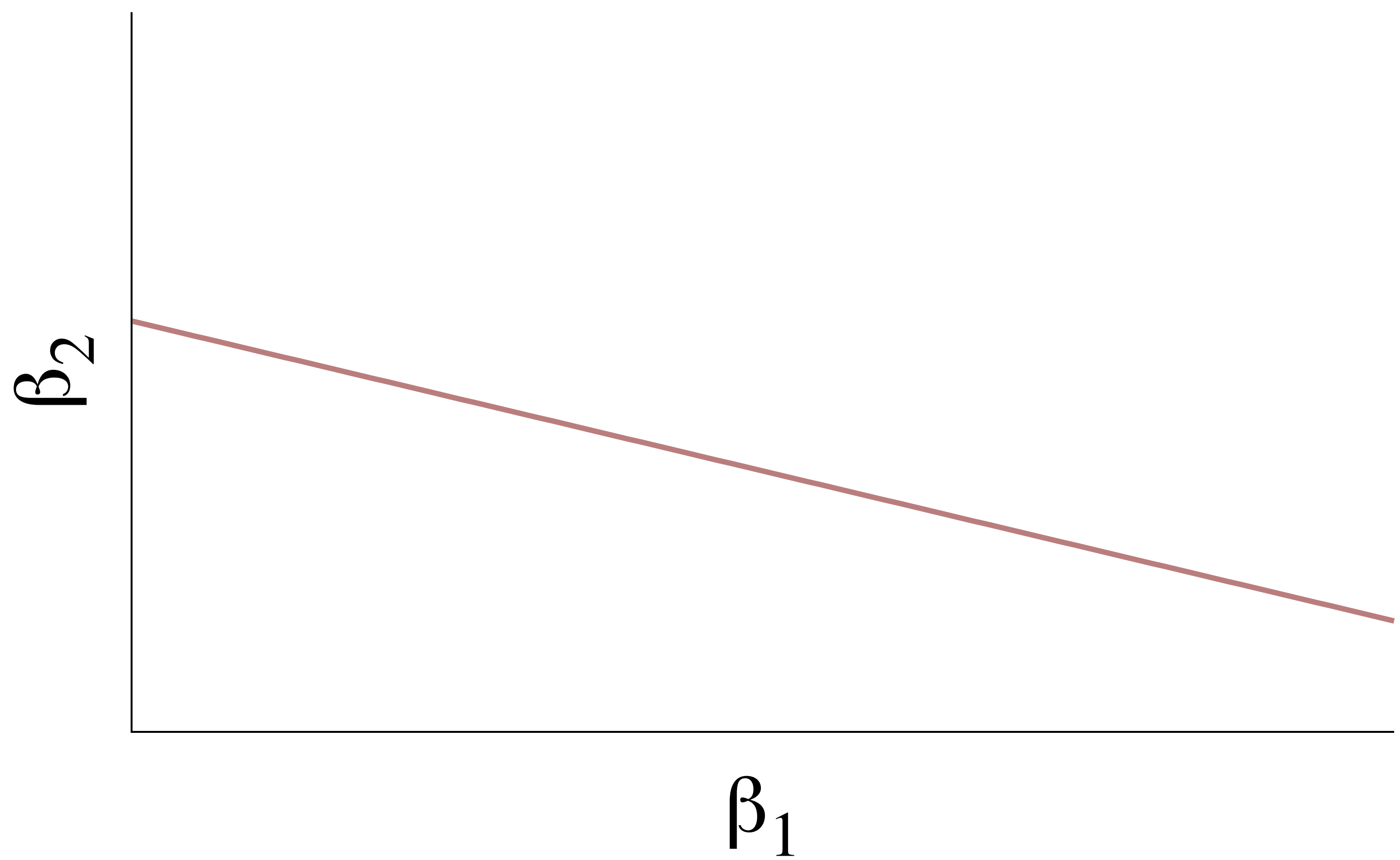
Given enough data, linear models are over-constrained and all the slopes can be fit well.



When there are fewer data than covariates, however, linear models are subject to collinearity.



With collinearity some of the slopes are fully determined while the others are completely undetermined.



Consequently (weakly) informative priors are critical for building robust linear models.

$$\boldsymbol{\beta} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Omega})$$

Consequently (weakly) informative priors are critical for building robust linear models.

$$\beta_i \sim \mathcal{N}(\mu_i, \omega_i)$$

Consequently (weakly) informative priors are critical for building robust linear models.

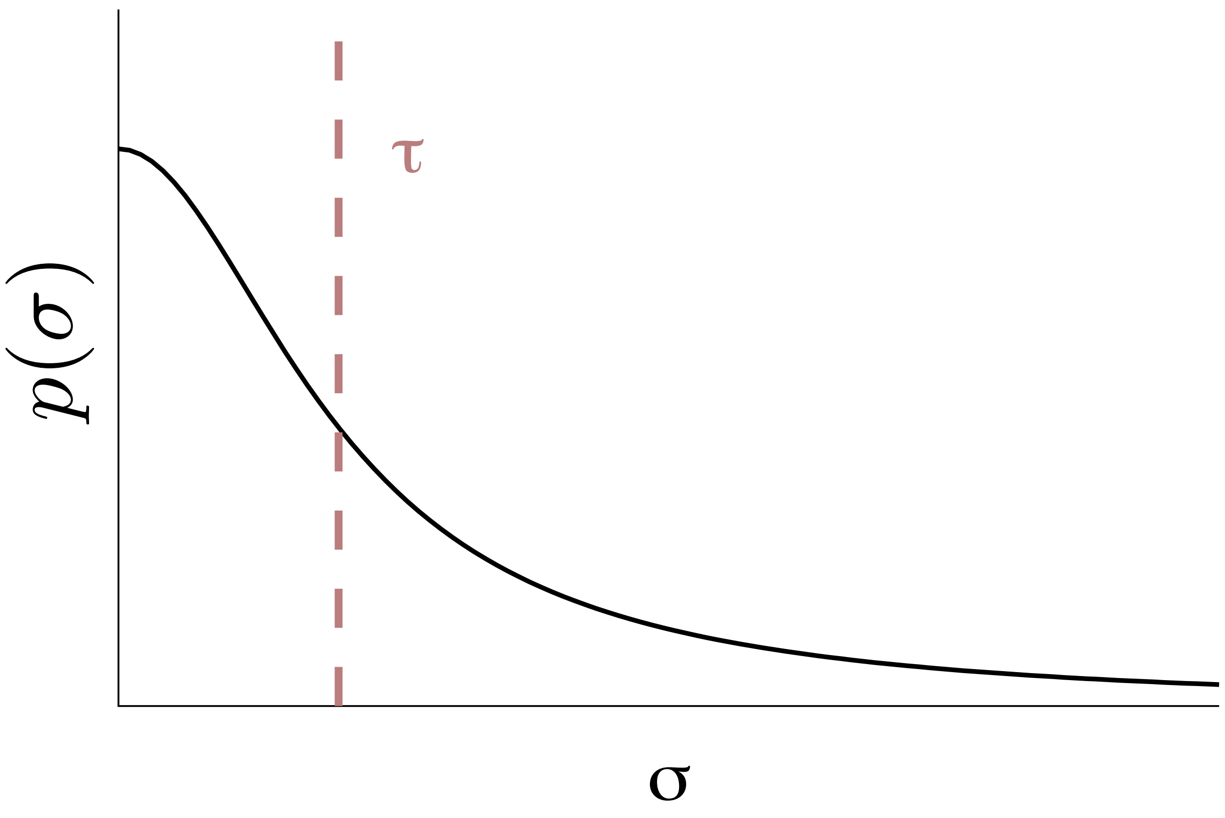
$$\beta_i \sim \mathcal{N}(0, \omega)$$

As with the linear model parameters, prior information for the Gaussian noise is critical.

$$p(\sigma) = \text{Half-Cauchy}(0, \tau)$$

As with the linear model parameters, prior information for the Gaussian noise is critical.

$$p(\sigma) = \text{Half-Cauchy}(0, \tau)$$



Gaussian-Linear model with proper priors

```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);



}
```

Gaussian-Linear model with proper priors

```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);
  beta ~ normal(0, 10);


}
```

Gaussian-Linear model with proper priors

```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);

}
```

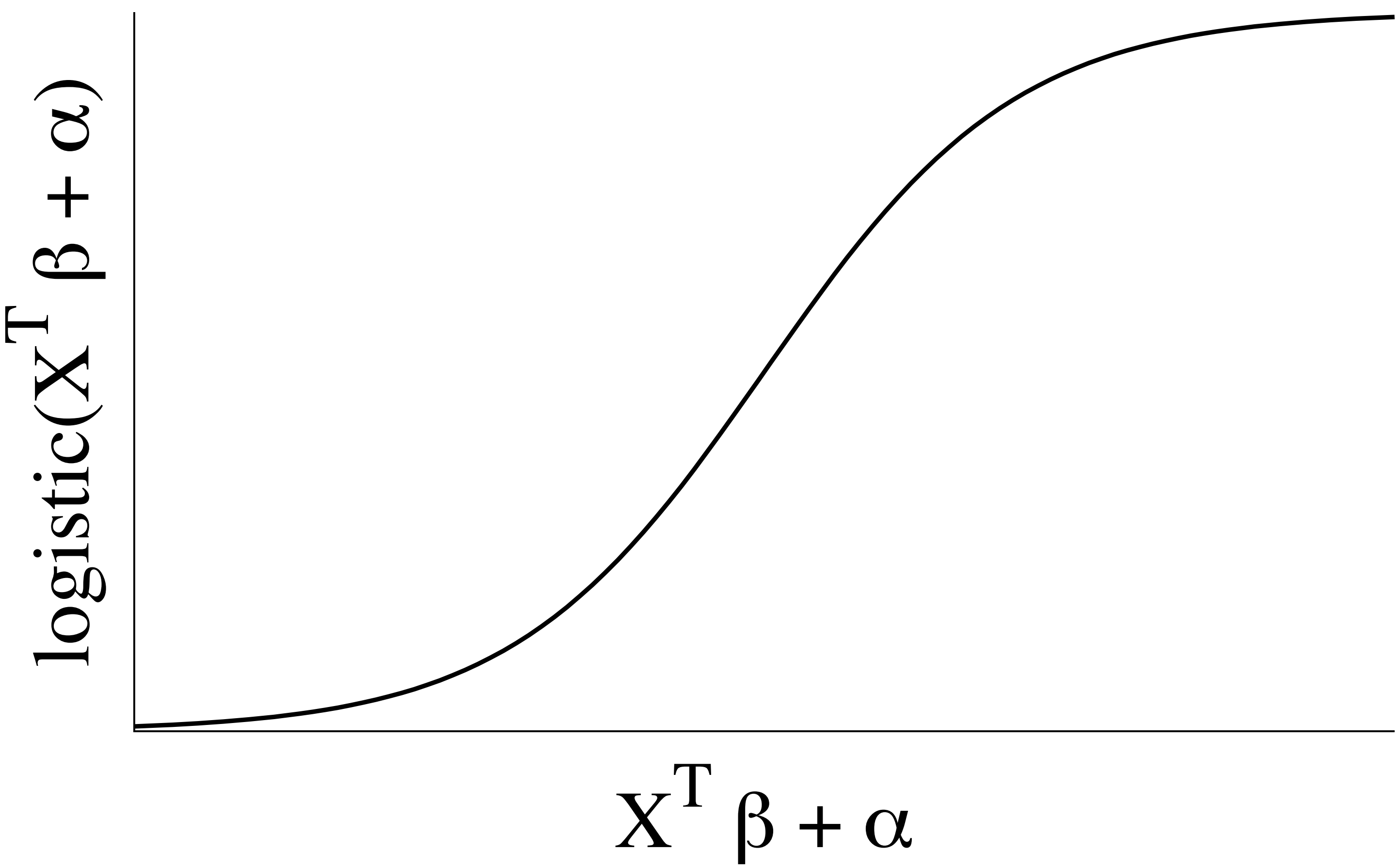
Gaussian-Linear model
with proper priors

```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  sigma ~ cauchy(0, 10);
}
```

Gaussian-Linear model with proper priors

```
data {
  int N;
  int K;
  real y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
  real<lower=0> sigma;
}
model {
  y ~ normal(X * beta + alpha, sigma);
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  sigma ~ cauchy(0, 10); // Half-Cauchy
}
```

Generalized Linear Models



Constrained parameters are
not amenable to linear models.

$$\theta \in (a, b)$$

$$\mathbf{X}^T \boldsymbol{\beta} + \alpha \in (-\infty, \infty)$$

We need to apply a transformation to the unconstrained linear predictor.

$$\theta \in (a, b)$$

$$g(\mathbf{X}^T \boldsymbol{\beta} + \alpha) \in (a, b)$$

In the statistics literature link functions are defined by the un-constraining map.

$$g^{-1} : (a, b) \to (-\infty, \infty)$$

Positive parameters are modeled with the *log* link function.

$$\log : (0, \infty) \to (-\infty, \infty)$$

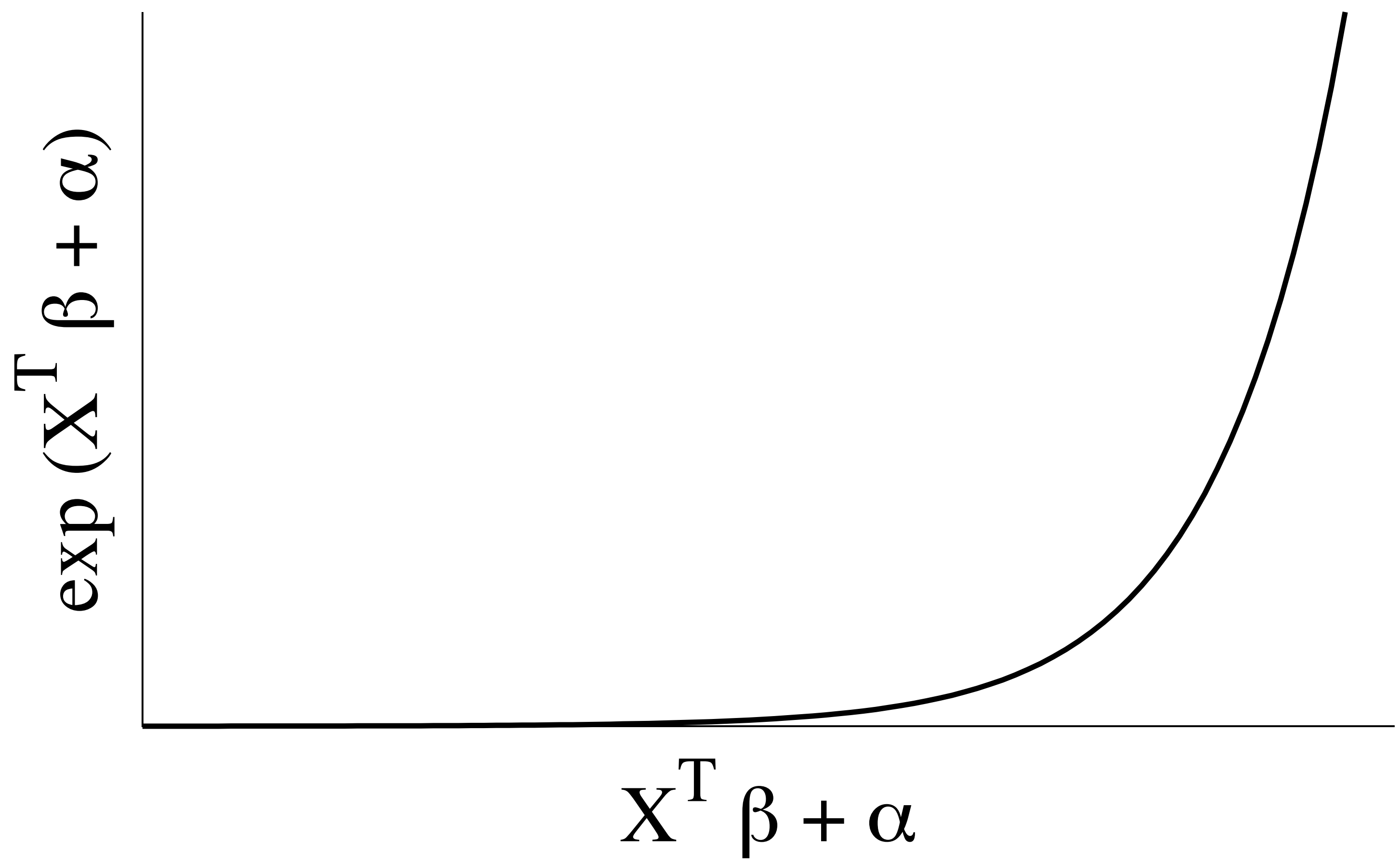
$$\exp\left(\mathbf{X}^T \boldsymbol{\beta} + \alpha\right) \in (0, \infty)$$

Positive parameters are modeled with the *log* link function.

link $$\log : (0, \infty) \to (-\infty, \infty)$$

inverse link $$\exp\left(\mathbf{X}^T\boldsymbol{\beta} + \alpha\right) \in (0, \infty)$$

Positive parameters are modeled with the *log* link function.



While bounded parameters are modeled with the *logit* link function.

$$\mathrm{logit} : (0,1) \to (-\infty, \infty)$$

$$\mathrm{logistic}\left(\mathbf{X}^T \boldsymbol{\beta} + \alpha\right) \in (0,1)$$

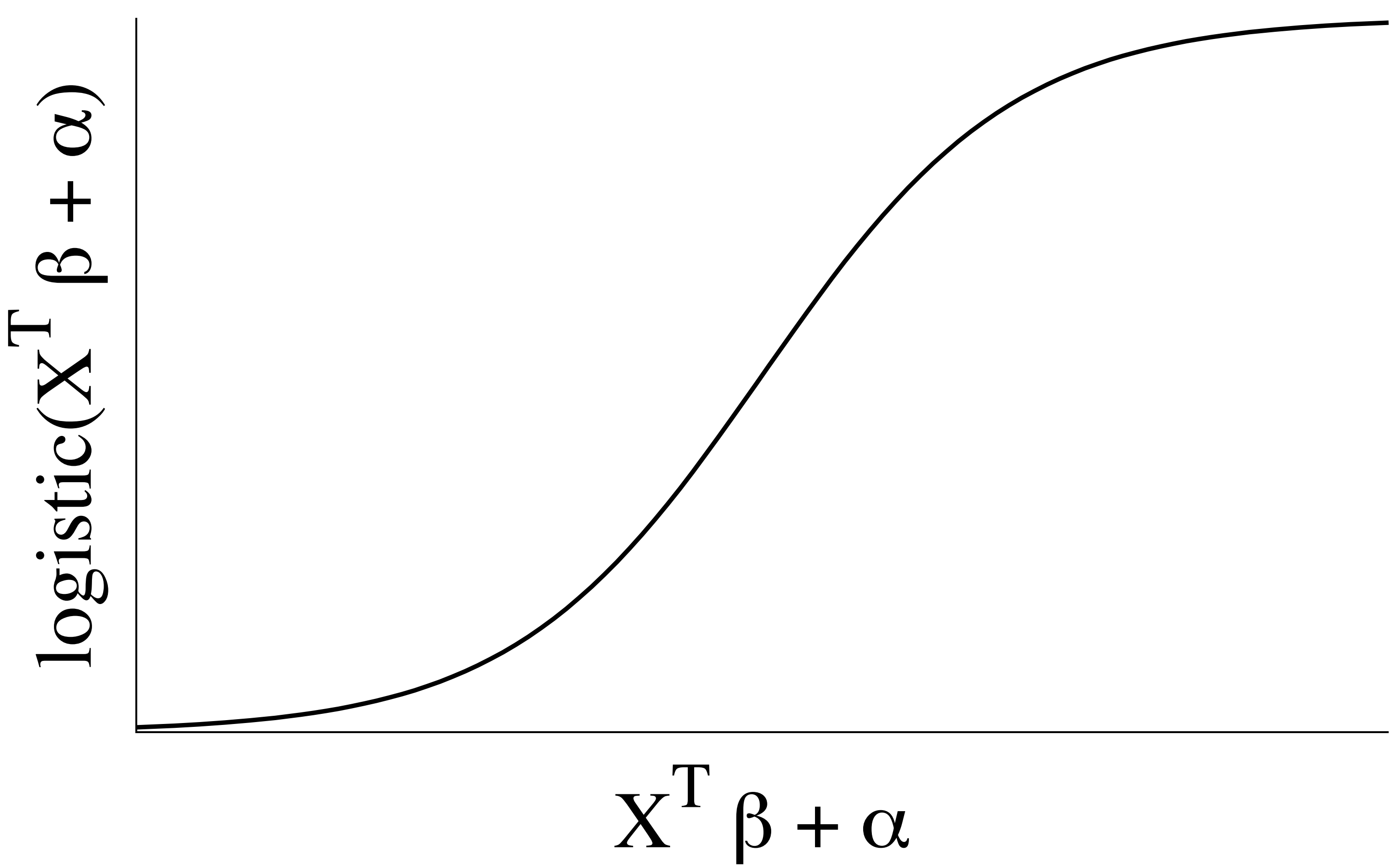
While bounded parameters are modeled with the *logit* link function.

link $$\text{logit} : (0, 1) \to (-\infty, \infty)$$

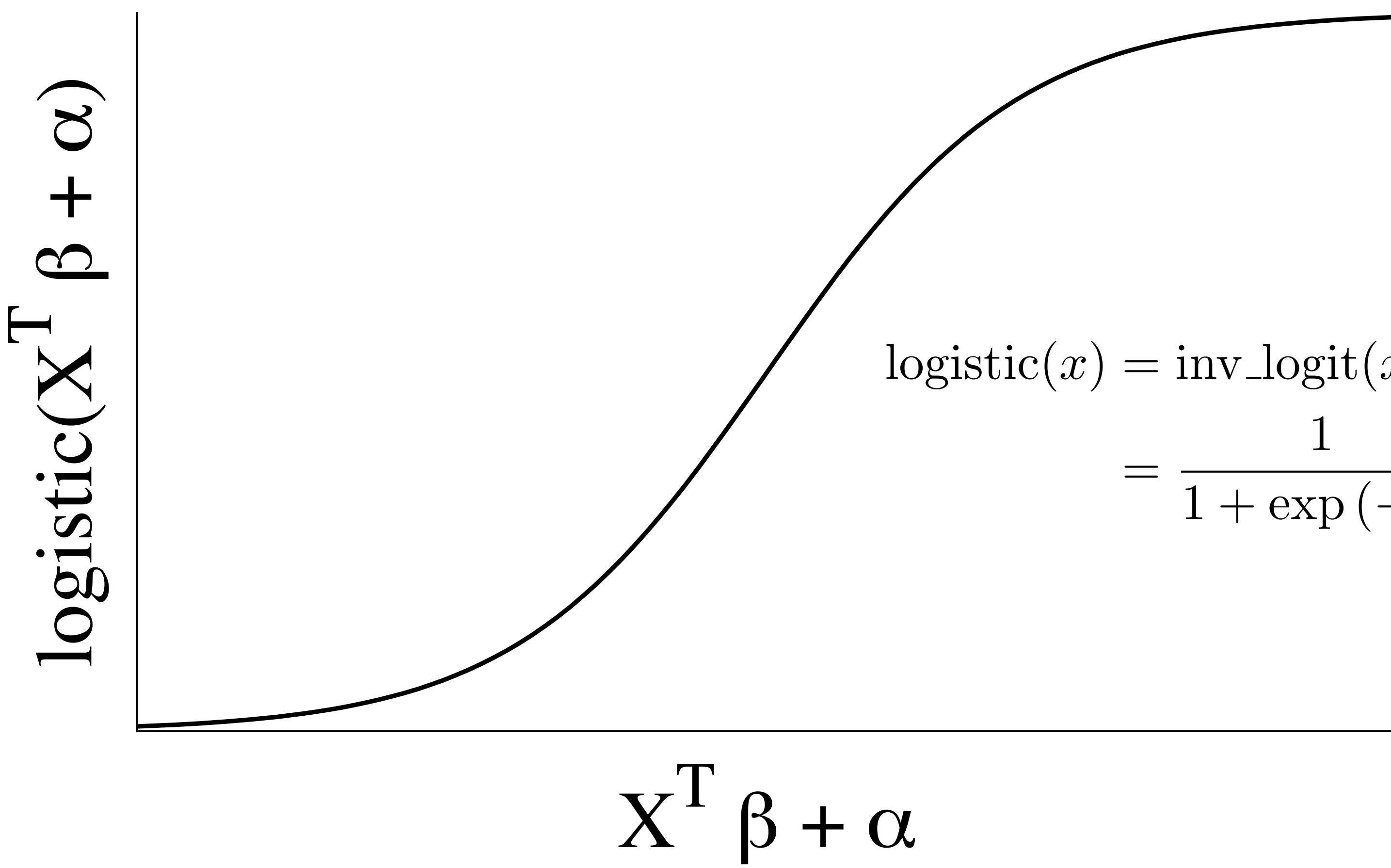
$$\text{logit}(x) = \log \frac{x}{1 - x}$$

inverse link $$\text{logistic}\left(\mathbf{X}^T \boldsymbol{\beta} + \alpha\right) \in (0, 1)$$

While bounded parameters are modeled with the *logit* link function.



While bounded parameters are modeled with the *logit* link function.



$$\text{logistic}(x) = \text{inv\_logit}(x) = \text{logit}^{-1}(x)$$
$$= \frac{1}{1 + \exp(-x)}$$

Success/failure data subject to covariates can be modeled with generalized binomial/Bernoulli models.

$$p(y|\mathbf{X}, \alpha, \boldsymbol{\beta}) = \mathrm{Ber}\big(y|\mathrm{logistic}\big(\mathbf{X}^T\boldsymbol{\beta} + \alpha\big)\big)$$

Success/failure data subject to covariates can be modeled with generalized binomial/Bernoulli models.

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli(inv_logit(alpha + X * beta));

}
```

Success/failure data subject to covariates can be modeled with generalized binomial/Bernoulli models.

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli(inv_logit(alpha + X * beta));

}
```

Success/failure data subject to covariates can be modeled with generalized binomial/Bernoulli models.

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli(inv_logit(alpha + X * beta));
  y ~ bernoulli_logit(alpha + X * beta);
}
```

How would you modify this code if each observation consists of multiple Bernoulli trials?

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;

}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli_logit(alpha + X * beta);

}
```

How would you modify this code if each observation consists of multiple Bernoulli trials?

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;
  int<lower=1> N_trials[N];
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli_logit(alpha + X * beta);

}
```

How would you modify this code if each observation consists of multiple Bernoulli trials?

```
data {
  int N;
  int K;
  int<lower=0,upper=1> y[N];
  matrix[N, K] X;
  int<lower=1> N_trials[N];
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ bernoulli_logit(alpha + X * beta);
  y ~ binomial_logit(N_trials, alpha + X * beta);
}
```

Count data whose rate depends on covariates can be modeled with a generalized Poisson model.

$$p(y|\mathbf{X}, \alpha, \boldsymbol{\beta}) = \mathrm{Poisson}\big(y|\exp\big(\mathbf{X}^T\boldsymbol{\beta} + \alpha\big)\big)$$

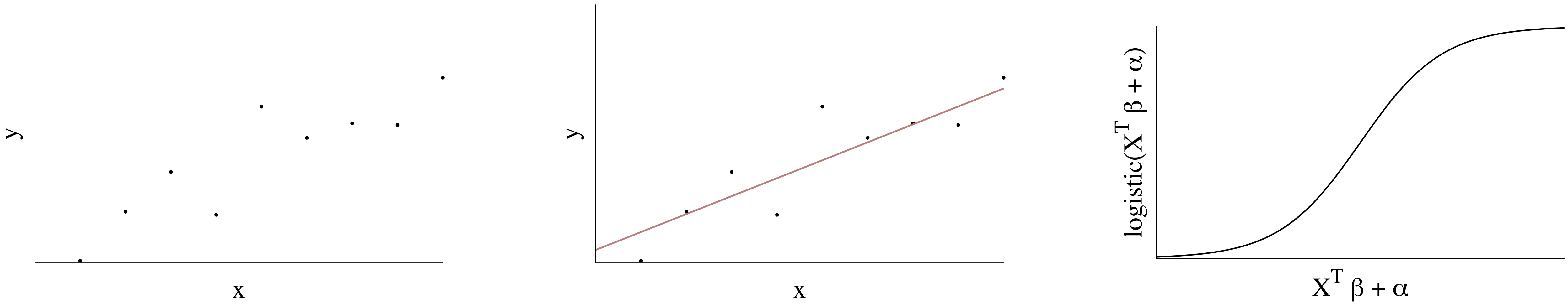
Count data with rate depending on covariates can be modeled with a generalized Poisson model.

```
data {
  int N;
  int K;
  int<lower=0> y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ poisson(exp(alpha + X * beta));

}
```

Count data with rate depending on covariates can be modeled with a generalized Poisson model.

```
data {
  int N;
  int K;
  int<lower=0> y[N];
  matrix[N, K] X;
}
parameters {
  vector[K] beta;
  real alpha;
}
model {
  beta ~ normal(0, 10);
  alpha ~ normal(0, 10);
  y ~ poisson(exp(alpha + X * beta));
  y ~ poisson_log(alpha + X * beta);
}
```


y
x
y
x
logistic(X^T β + α)
X^T β + α