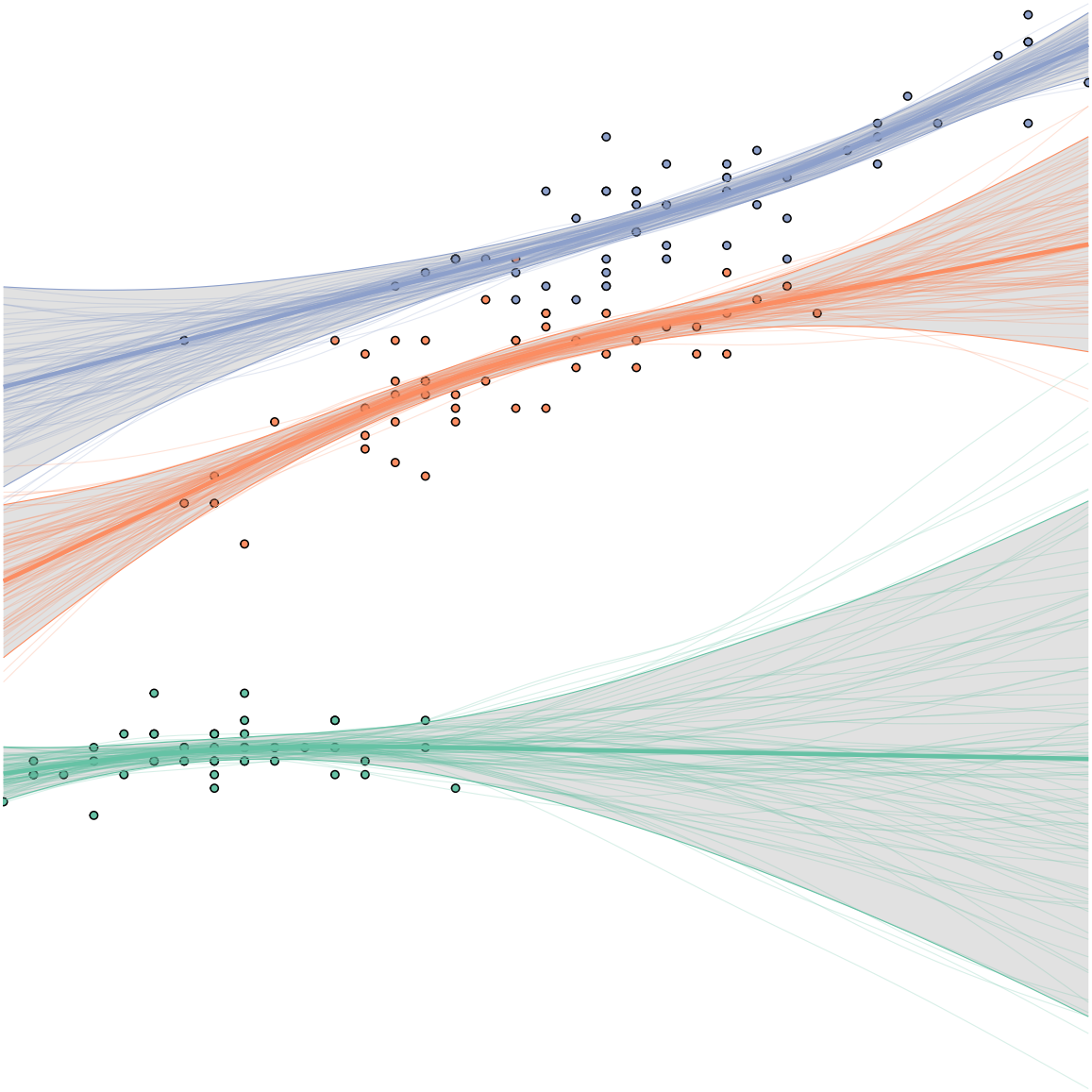
Why should I ever use GAMs when I just want a (G)LM?

Asger Svenning[1]

[1]Department of Ecoscience, Aarhus University (asgersvenning@ecos.au.dk)



Introduction

In this short example I hope to convince you that `mgcv` and the `gam` function is useful for more than just GAMs, but can - and indeed should - often be used in the case where the relationship under interest is linear (i.e. where a LM would usually be used).

I will show how an unknown non-linear confounding variable can be accounted for using a GAM, while including or excluding it in a standard LM leads to biased estimates, potentially resulting in opposite conclusions.

Sampling a smooth confounding curve

To illustrate the problem, we will first need to define a way to generate a non-linear function. Later we will sample a new random non-linear function for each experiment, which is then used as the relationship between the confounding variable and the response.

To do this, we generate N evenly spaced control points, c, along the z-axis, and then sample N random values from a uniform distribution to generate the s-values. The position of the control points on the z-axis is then subtracted from the s-values to give a slight overall negative trend on average:

$$\begin{aligned} c_z &\sim \mathcal{U}(z_{\min}, z_{\max}), & z_{\min} &= -0.1, & z_{\max} &= 1.1 \\ c_s &\sim \mathcal{U}(s_{\min}, s_{\max}) - z, & s_{\min} &= -1, & s_{\max} &= 1 \end{aligned}$$

This is done to shift the distribution of confounding functions in such a way that the confounding effect is more likely to be opposite the true effect of x on y, which for this experiment will be positive one-to-one relationship.

We then use the `splinefun` function to generate a function, f_s, that smoothly interpolates between the control points:

$$f_s(z) = \text{splinefun}(c_z, c_s, \text{method} = \text{"natural"})$$

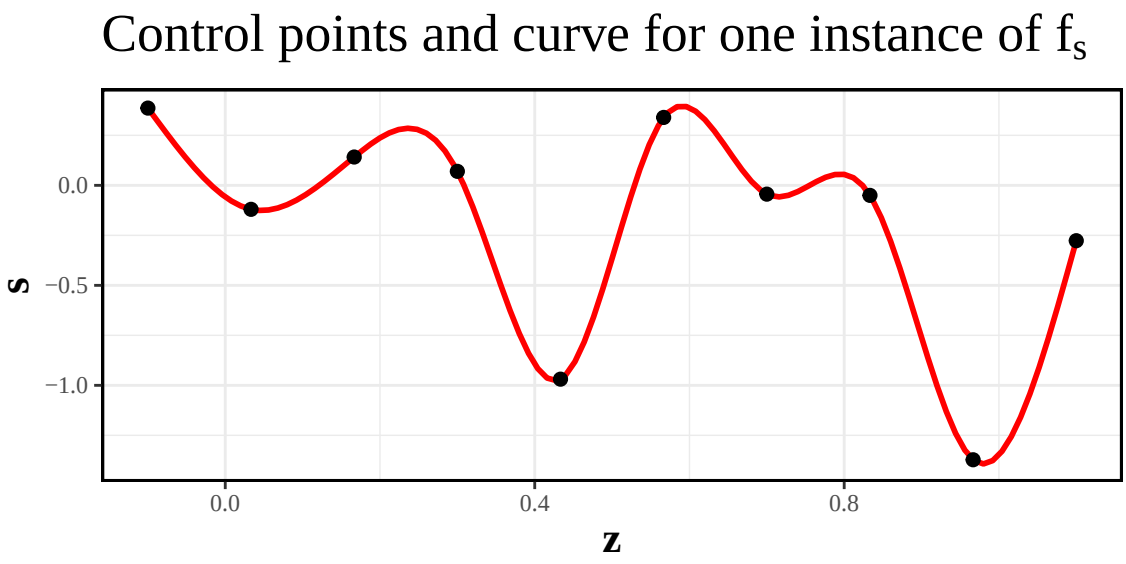
```
get_sfun <- function(zlim = c(-0.1, 1.1), slim = c(-1, 1), ctrl.pts = 10, plot = F) {
  z <- seq(zlim[1], zlim[2], length.out = ctrl.pts)
  c <- runif(ctrl.pts, min = slim[1], max = slim[2]) - z

  sfun <- splinefun(z, c, method = "natural")
  if (plot) plot_smooth(sfun, z, c, zlim)

  return(invisible(sfun))
}
```

Example smooth confounding curve

The following plot shows an example of a smooth confounding curve generated using the `get_sfun` function.



The experiment

We will now define a simple experiment where we generate a dataset with three variables: x, z, and y. Both x and z are generated as linear functions of a latent variable l, with a small amount of uniform noise added. This results in x and z being highly correlated (correlation coefficient $\rho \approx 0.95$), leading to multicollinearity when both are included in a regression model.

Multicollinearity and Confounding Variables

Multicollinearity occurs when predictor variables are highly correlated, which can inflate the variance of the coefficient estimates and make them unstable. A common approach in some introductory statistics courses or guides is to remove variables with high Variance Inflation Factor (VIF) values to mitigate multicollinearity. For instance, variables with a VIF greater than 5 or 10 are sometimes considered problematic and candidates for removal.

However, when the correlated variable is a confounder—meaning it influences both the predictor and the response—excluding it from the model can introduce bias due to omitted variable bias. In our case, z is a confounding variable affecting both x and y. Removing z to reduce multicollinearity ignores its confounding effect, potentially leading to biased and inconsistent estimates of the effect of x on y.

While high multicollinearity can be concerning, it's important to weigh the trade-offs. Including the confounding variable ensures that we account for its effect, obtaining unbiased estimates of the primary predictor. In situations where the goal is to make accurate inferences about the relationships between variables, it's generally advisable to include confounders in the model, even at the expense of increased multicollinearity.

In our experiment, the VIF for x and z is around 13–14, exceeding common thresholds. Despite this, we will include both variables in our models to highlight the importance of accounting for confounding effects, and we will explore how GAMs can handle this situation effectively.

Predictor-Reponse Relationship

The response variable y is then generated as a linear function of x, a randomly sampled non-linear function of $f_s(z)$, and some normally distributed noise:

$$y = x + 3 \cdot f_s(z) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, 1/2)$$

The specific form of the predictor-response relationship was chosen to ensure that the effect of the confounding variable, z, is stronger than the effect of the predictor of interest, x, while the noise is kept small enough that residual patterns in the diagnostic plots are very clear.

Model Fitting

We then fit three models to the data:

- A GAM model with y as a function of x and a smooth term for z: `gam(y ~ x + s(z))`.
- A linear model with y as a function of x and z: `lm(y ~ x + z)`.
- A linear model with y as a function of x only: `lm(y ~ x)`.

The `experiment` function below generates the data, fits the models, and returns the estimated coefficient for x and the standard error for each model. We also calculate the error (the difference between the estimated coefficient and the true value of 1) and the standardized estimate error, calculated as:

$$z_{error} = \frac{\hat{\mu}_x - \mu_x}{\hat{\sigma}_{\hat{\mu}_x}}$$

where $\hat{\mu}x$ is the estimated coefficient for x, and $\hat{\sigma}\hat{\mu}_x$ is its standard error. Under the assumption that the estimator is unbiased and normally distributed, this standardized estimate error should follow a standard normal distribution (mean 0, standard deviation 1). This allows us to assess whether the deviations of the estimates from the true value are within the expected range due to sampling variability.

Note: It is very important to take notice of the fact that f_s is a random non-linear function, and that for each iteration of the experiment a new f_s is sampled. This is done to ensure that the results hold in general for non-linear confounding variables, and not just for a specific shape of f_s.

Experiment code

```r
experiment <- function(plot=F) {
  data <- tibble(
    # Sample the latent variable
    l = runif(500),
    # Generate x and z as linear functions of l with a small amount of
    # independent uniform noise to ensure they are highly, but not
    # perfectly, collinear
    x = l + runif(500, -1, 1) / 10,
    z = l + runif(500, -1, 1) / 10,
    # Generate y as a linear function of x and a sampled non-linear function
    # of z, with some normally distributed noise
    y = x + 5 * get_sfun()(z) + rnorm(500, 0, 1/2)
  )

  if (plot) dp <- plot_data(data)

  # Fit the three models
  gam_mod <- bam(y ~ x + s(z), data = data, method = "fREML", discrete=T)
  lin_mod1 <- lm(y ~ x + z, data = data)
  lin_mod2 <- lm(y ~ x, data = data)

  if (plot) plot_models(gam_mod, lin_mod1, lin_mod2, dp)

  # Extract and summarize the results for each model
  res <- tibble(
    type = c("gam", "lin1", "lin2"),
    model = list(gam_mod, lin_mod1, lin_mod2),
    result = map(model, ~ broom::tidy(.x, parametric=T))
  ) %>%
    select(!model) %>%
    unnest(result) %>%
    select(type, term, estimate, std.error) %>%
    filter(term == "x") %>%
    mutate(
      error = estimate - 1,
      z.error = error / std.error
    )

  return(invisible(res))
}
```

Example diagnostics

Before we dive into the results, let's look at the diagnostics for the three models fitted to a particular instantiation of the data.

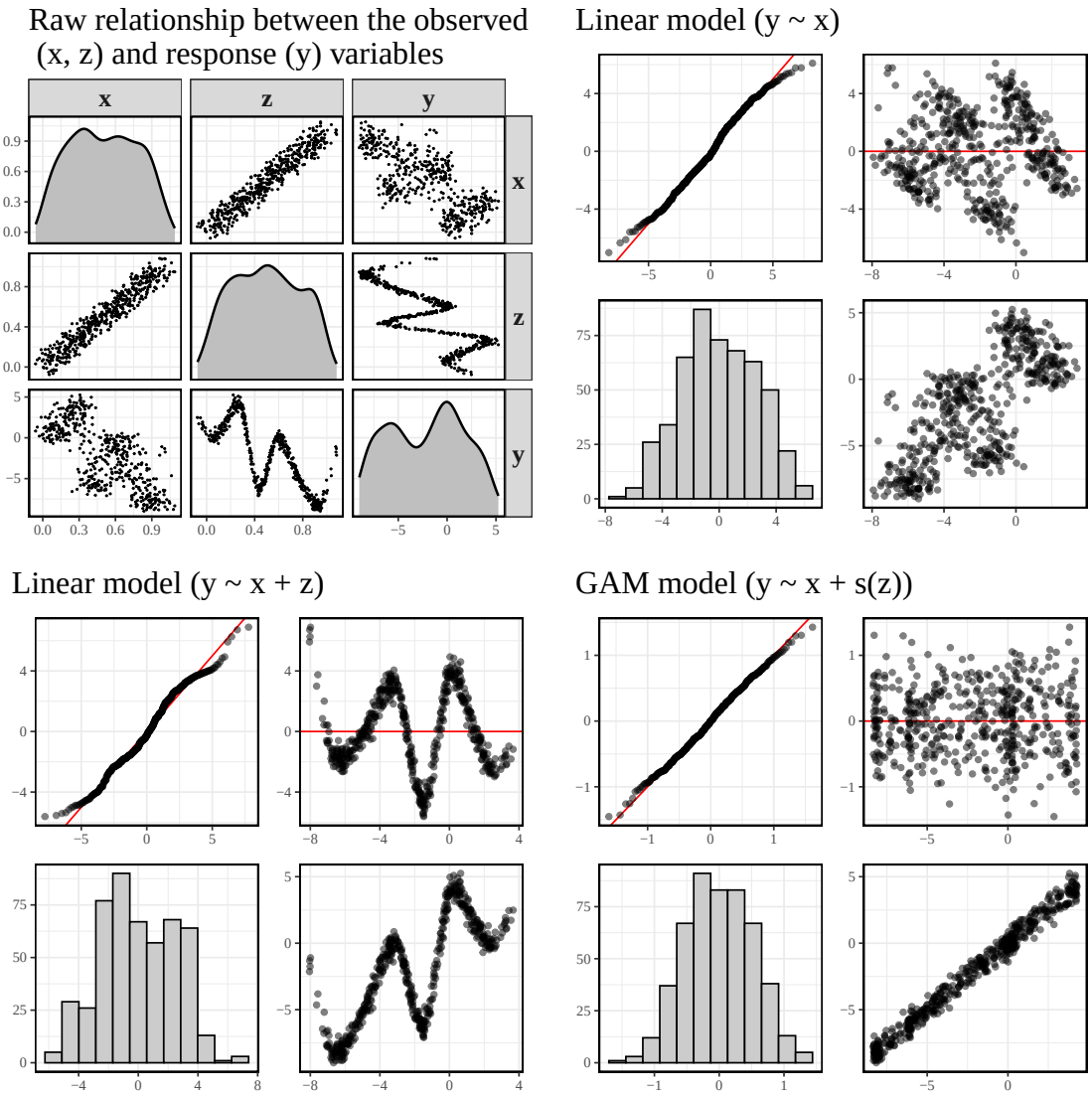


Figure 1: (TR, BL & BR) Diagnostic plots for the three models with sub-panels; QQ-plot (TL), residuals-fitted/linear predictor (TR), residual histogram (BL) and observed-fitted (BR). (TL) Pairwise scatter plot and histograms for x, y and z.

As can be clearly seen by inspecting the diagnostic plots, the GAM model captures the relationship between y and z well, leading to acceptable model diagnostics, whereas the linear models would have to be rejected based on the diagnostics. However, does this also result in incorrect estimates of the coefficient for x (μ_x)?

Results

To answer the question about whether the GAM model leads to better estimates of the coefficient for x, we will run the `experiment` function 2000 times and compare the results for the three models. Remember that for each iteration we sample a new smooth function f_s and generate new data, this is done to ensure that the results hold in general for non-linear confounding variables.

Summary of results

The table below shows the average estimated coefficient for x ($\widetilde{\mu_x}$), the mean absolute error ($\overline{|\widetilde{\mu_x} - \mu_x|}$), the rate of confident opposite conclusions ($\text{sign}(\widetilde{\mu_x}) \neq \text{sign}(\mu_x) \wedge |\tilde{\mu}_x / \widetilde{\sigma(\mu_x)}| > 1.96$), and the rate of estimates within the confidence interval ($|\widetilde{\mu_x} - \mu_x| / \widetilde{\sigma(\mu_x)} < 1.96$) for the three models.

Model	Mean estimated slope	Mean absolute error	Rate of confident opposite conclusions	Rate of estimate within confidence interval
$y \sim x + s(z)$	1.00	0.26	0.00%	94.75%
$y \sim x + z$	0.94	1.58	4.35%	78.80%
$y \sim x$	-3.96	5.11	84.60%	5.00%

The three models also differ in the rate of rejecting the null hypothesis of $\mu_x = 0$, with the GAM model rejecting the null hypothesis in 87% of the cases, the linear model with $y \sim x + z$ rejecting the null hypothesis in 26% of the cases, and the linear model with $y \sim x$ rejecting the null hypothesis in 92% of the cases.

Expected versus observed error distribution

The following plot shows the distribution of the standardized estimate error (z_{error}) for the three models. The blue line represents the expected standard normal distribution under the assumption that our estimator is unbiased and the model is correctly specified. Deviations from this distribution indicate potential bias or model misspecification.

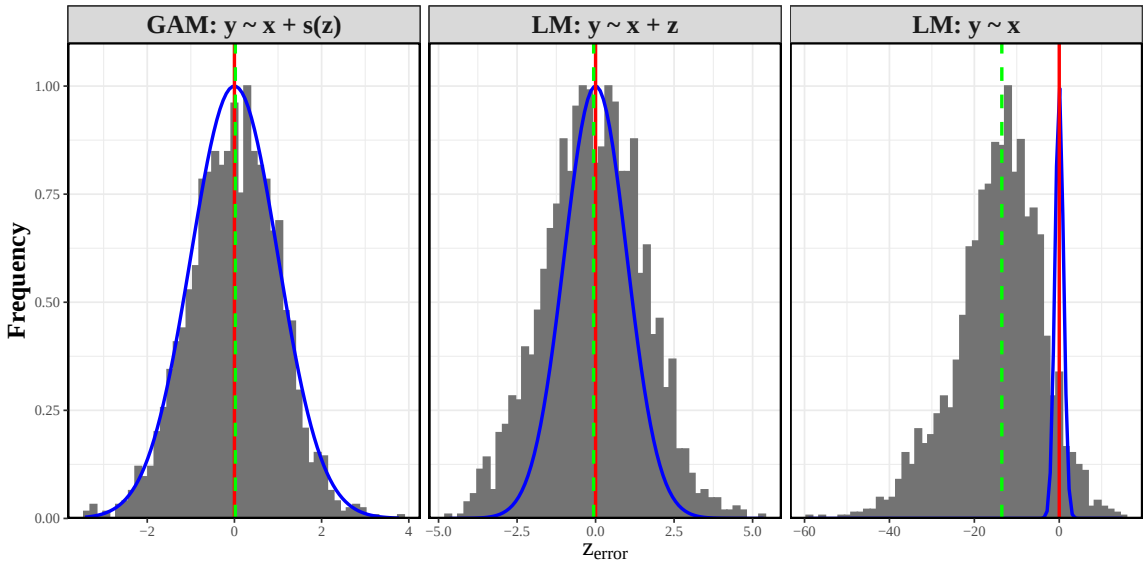


Figure 2: Distribution of the standardized estimate error (z_{error}) for the three models. The blue curve represents the expected standard normal distribution (mean 0, standard deviation 1). The green dashed line indicates the median z_{error}-score for each model, and the red line marks the expected mean of zero under the assumption of unbiased estimates.

Conclusion

This simple example highlights the power of Generalized Additive Models (GAMs) in accounting for unknown non-linear relationships, particularly with confounding variables. Even when our primary interest lies in modeling a linear relationship, as with y and x, ignoring or improperly handling confounders like z can lead to biased estimates and misleading conclusions.

In our simulations, the GAM consistently provided unbiased estimates of the effect of x on y, while the linear models struggled, especially when z was excluded or included without accounting for its non-linear relationship with y. This demonstrates that GAMs are not just tools for modeling non-linear primary effects but are also invaluable for adjusting for non-linear confounding effects.

Key Takeaways

- **Importance of Including Confounders**: Always consider including potential confounding variables in your models, even if they are highly correlated with your predictors. Excluding them can introduce significant bias.
- **Handling Multicollinearity**: High multicollinearity increases the variance of coefficient estimates but does not bias them. When dealing with confounders, it's often better to accept higher variance to maintain unbiasedness.
- **Flexibility of GAMs**: GAMs offer a flexible framework to model complex, non-linear relationships without specifying a particular functional form. This makes them particularly useful in exploratory analyses and when dealing with unknown or complex confounding effects.
- **Practical Application**: In real-world data analysis, especially with observational data, relationships are often non-linear especially for confounding variables. GAMs provide a robust tool for uncovering and adjusting for these complexities, leading to more reliable and insightful results.