STAT410 Assignment: Advanced Reading Topic

Ben Murarotto

May 2025

Contents

1 Introduction

Inflation is a hot topic in the current global economic climate, and is important for the impact it has on many factors of the economy, including interest rates which are impacting household budgets (directly and indirectly) significantly. The data in the file inflation.csv contains annual inflation data for 5 developed nations for the period of 1960 to 2023.

There are three variables in the data set: - **Country**: Country identifier. - **Year**: year inflation was recorded. - **Inflation**: annual inflation (%).

The research question is: *Is there evidence that yearly inflation data is changing over time? If so, how is it changing?*

1.1 Exploring the dataset.

```r
inf.df <- read.csv("inflation.csv", header = T)
library(ggplot2)

head(inf.df, 5)
```

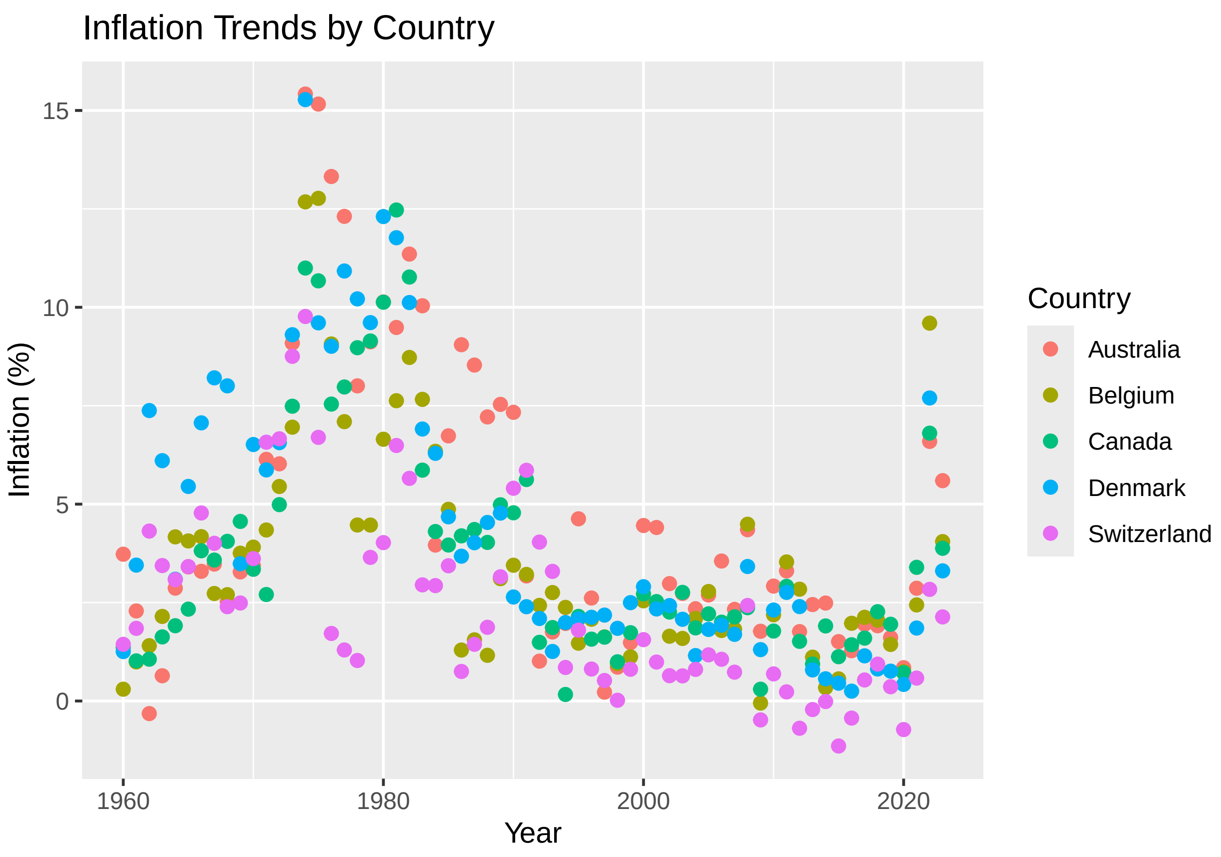
```
##       Country Year Inflation
## 1   Australia 1960 3.7288136
## 2     Belgium 1960 0.2994673
## 3      Canada 1960 1.3586957
## 4 Switzerland 1960 1.4387947
## 5     Denmark 1960 1.2552301
```

We note that the 5 countries in the data set are Australia, Belgium, Canada, Denmark and Switzerland. Lets create a subset for each country.

```r
countries = c("Australia", "Belgium", "Canada", "Denmark", "Switzerland")
dataframes = list()

for (country in countries){
   dataframes[[country]] <- subset(inf.df, Country == country)
}
```

1.2 Plotting.

```r
p <- ggplot() +
  labs(title = "Inflation Trends by Country",
       x = "Year",
       y = "Inflation (%)")
for (i in 1:5){
  df <- dataframes[[i]]
  p <- p + geom_point(data = df,
                      aes(x = Year, y = Inflation, color = Country),
                      size = 2)
}
print(p)
```



Looking at the scatter data for inflation for each country we can see global trends at play. We see that the inflation percentage is not constant and follows a polynomial trend, peaking in the 1980's then steadily declining only to sharp rise again after 2020. This answer's the first part of our research question that yes, there is evidence of inflation changing over time. Our next objective will be to attempt to quantify this trend.

1.3 Testing models.

Let's use cross validation to compare degree's of polynomials to find the best fitting function for the data. To do this we are going to take advantage of the boot library.

```r
library(boot)
set.seed(66)
df01 <- subset(inf.df, select = -Country)
df01 <- na.omit(df01)
degrees <- 1:5
cv_error <- numeric(length(degrees))
models =

for (i in degrees){
  model <- glm(Inflation ~ poly(Year, i, raw = FALSE), data = df01, family = "gaussian")
  cv_result <- cv.glm(df01, model, K = 10)
  cv_error[i] <- cv_result$delta[1]
}

cv_error
```

```
## [1] 8.440461 8.279213 5.357456 5.193777 5.195786
```

```r
best_degree <- which.min(cv_error)
cat("Best polynomial degree:", best_degree, "with CV error:", min(cv_error), "\n")
```

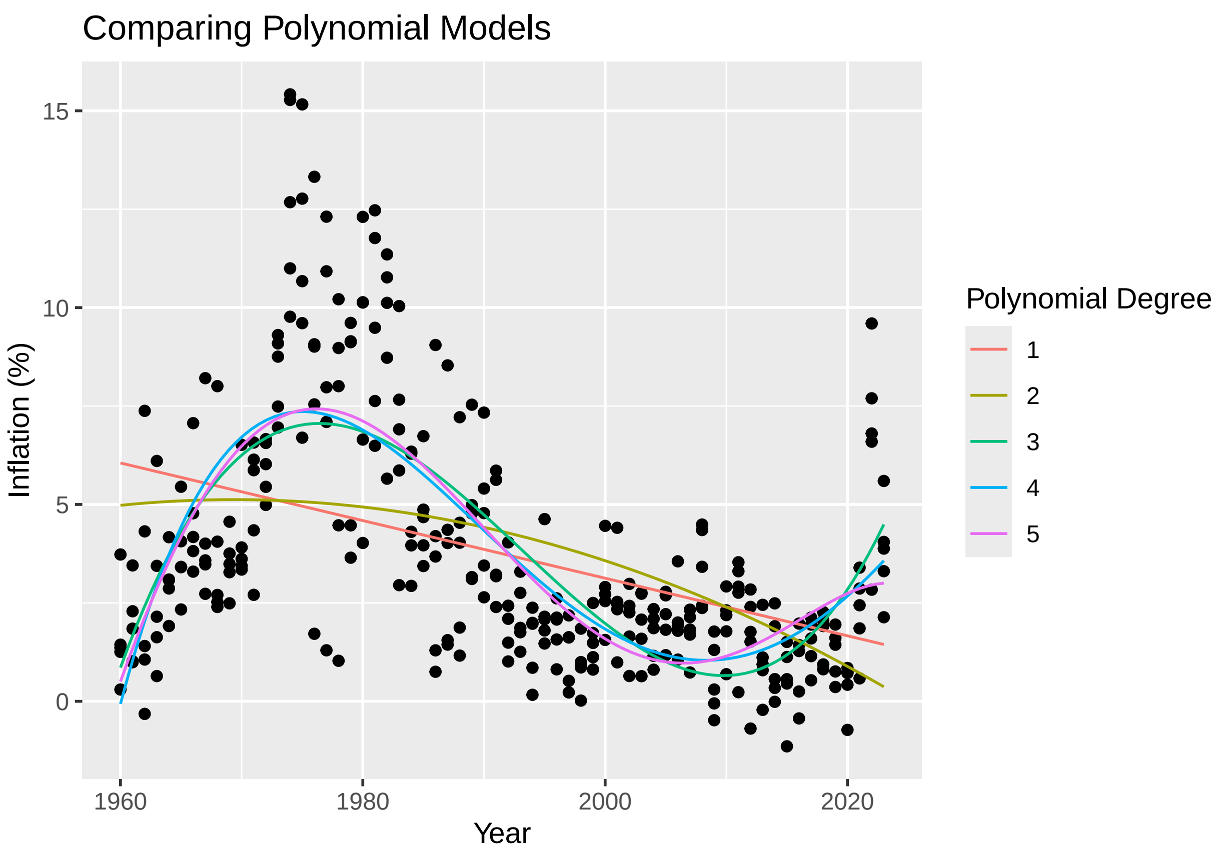
```
## Best polynomial degree: 4 with CV error: 5.193777
```

```r
library(ggplot2)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```r
newdata <- data.frame(Year = seq(min(df01$Year), max(df01$Year), length.out = 100))

all_preds <- data.frame()

for (i in 1:5) {
  model <- glm(Inflation ~ poly(Year, i, raw = FALSE), data = df01, family = "gaussian")
  preds <- predict(model, newdata = newdata)
  temp <- data.frame(Year = newdata$Year,
                     Prediction = preds,
                     Degree = factor(i))
  all_preds <- bind_rows(all_preds, temp)
}
```

```
ggplot(df01, aes(x = Year, y = Inflation)) +
  geom_point() +
  geom_line(data = all_preds, aes(x = Year, y = Prediction, color = Degree), linewidth = 0.5) +
  labs(title = "Comparing Polynomial Models",
       x = "Year",
       y = "Inflation (%)",
       color = "Polynomial Degree")
```



As we can see, with cross validation the 4th degree polynomial had the fit with least error. Let's compare AIC and BIC for further confirmation.

```
fit3 <- glm(Inflation ~ poly(Year, 3, raw = FALSE), data = df01, family = "gaussian")
fit4 <- glm(Inflation ~ poly(Year, 4, raw = FALSE), data = df01, family = "gaussian")
fit5 <- glm(Inflation ~ poly(Year, 5, raw = FALSE), data = df01, family = "gaussian")
```

```
AIC(fit3, fit4, fit5)
```

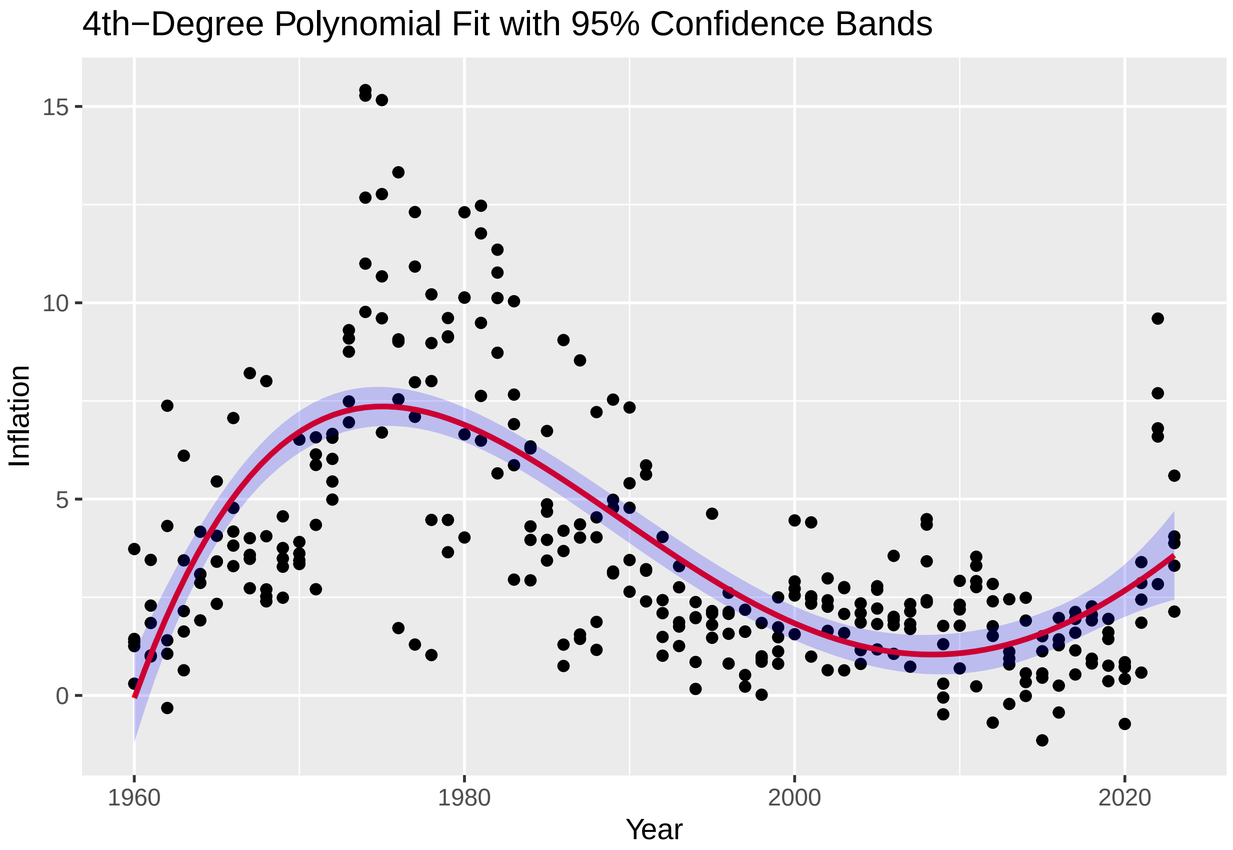
```
##      df      AIC
## fit3  5 1442.973
## fit4  6 1436.850
## fit5  7 1435.840
```

```
BIC(fit3, fit4, fit5)
```

```
##      df      BIC
## fit3  5 1461.814
## fit4  6 1459.459
## fit5  7 1462.218
```

BIC prefers fit 4, which is the more conservative test for complexity costs. Lets plot our final polynomial and 95% CI.

```r
preds <- predict(fit4, newdata = newdata, se.fit = TRUE)
preds_df <- data.frame(
  Year = newdata$Year,
  fit = preds$fit,
  lwr = preds$fit - 1.96 * preds$se.fit,
  upr = preds$fit + 1.96 * preds$se.fit
)

ggplot() +
  geom_point(data = df01, aes(x = Year, y = Inflation)) +
  geom_line(data = preds_df, aes(x = Year, y = fit), color = "red", linewidth = 1) +
  geom_ribbon(
    data = preds_df,
    aes(x = Year, ymin = lwr, ymax = upr),
    fill = "blue", alpha = 0.2
  ) +
  labs(
    title = "4th-Degree Polynomial Fit with 95% Confidence Bands",
    x = "Year",
    y = "Inflation"
  )
```

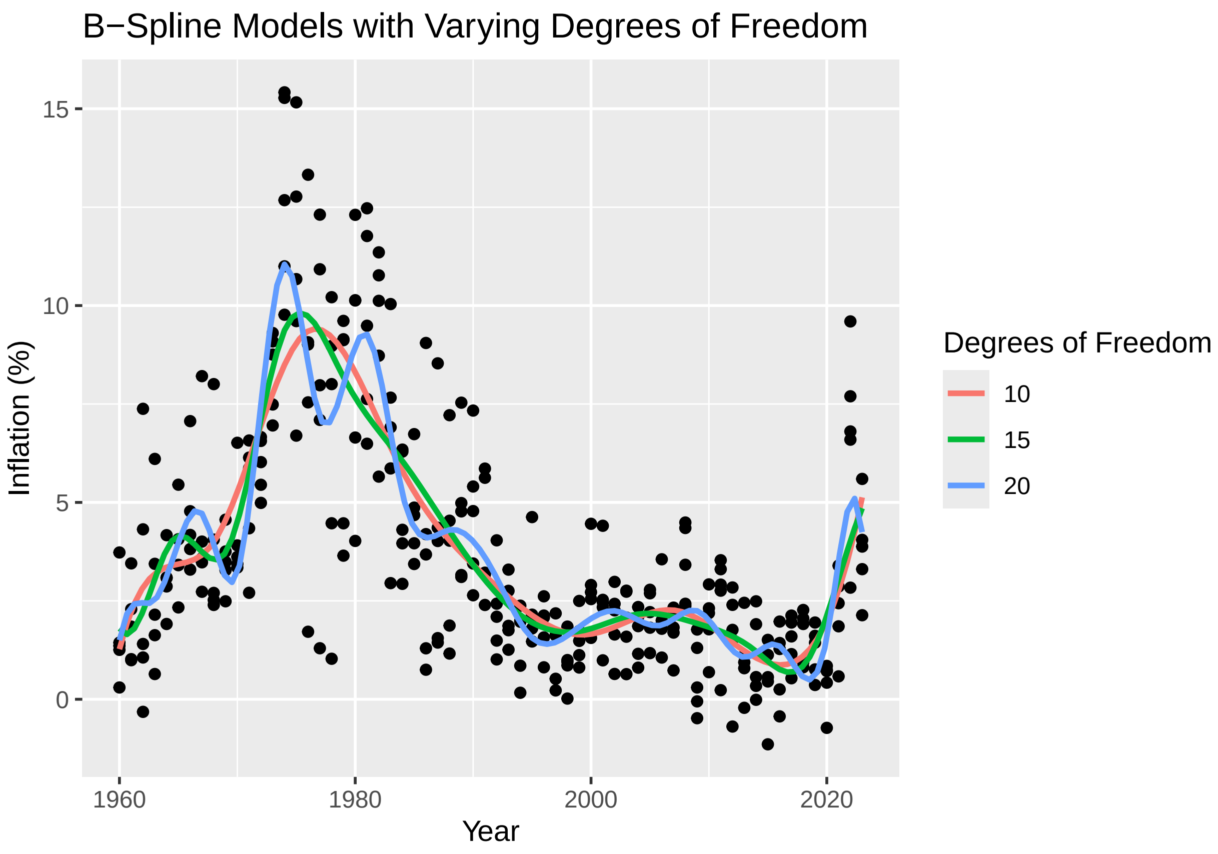


2 Fitting Non-Parametric Functions.

Our 4th degree polynomial function captures the general trend of the data, however we can test a range of non parametric functions to estimate the effects in the localised regions that are missed by the polynomial function such as the peak in inflation around the 1980's.

2.1 Fitting B-Spline

```r
library(splines)
bspline_preds <- data.frame()

for (i in c(10, 15, 20)) {
  model.bs <- lm(Inflation ~ bs(Year, df = i), data = df01)
  preds.bs <- predict(model.bs, newdata = newdata, se = TRUE)
  temp <- data.frame(
    Year = newdata$Year,
    Prediction = preds.bs$fit,
    DF = factor(i)
  )
  bspline_preds <- bind_rows(bspline_preds, temp)
}

ggplot(df01, aes(x = Year, y = Inflation)) +
  geom_point() +
  geom_line(
    data = bspline_preds,
    aes(x = Year, y = Prediction, color = DF),
    linewidth = 1
  ) +
  labs(
    title = "B-Spline Models with Varying Degrees of Freedom",
    x = "Year",
    y = "Inflation (%)",
    color = "Degrees of Freedom"
  )
```



Since our objective is finding best fit and not to extrapolate, we can lean towards greater degrees of freedom in our B-spline. When this metric is raised the model smooths less and emphasises local effects.

Let's save our model with 14df to use later.

```
final_blspine <- lm(Inflation ~ bs(Year, df = 14), data = df01)
```

2.2 Fitting a kernel smoother.

The advanced reading for this topic discusses the Epanechnikov Kernel as a sensible choice for in terms of smoothness and performance. The np library uses CV to automatically select the ideal bandwidth for the data.

```
library(np)
```

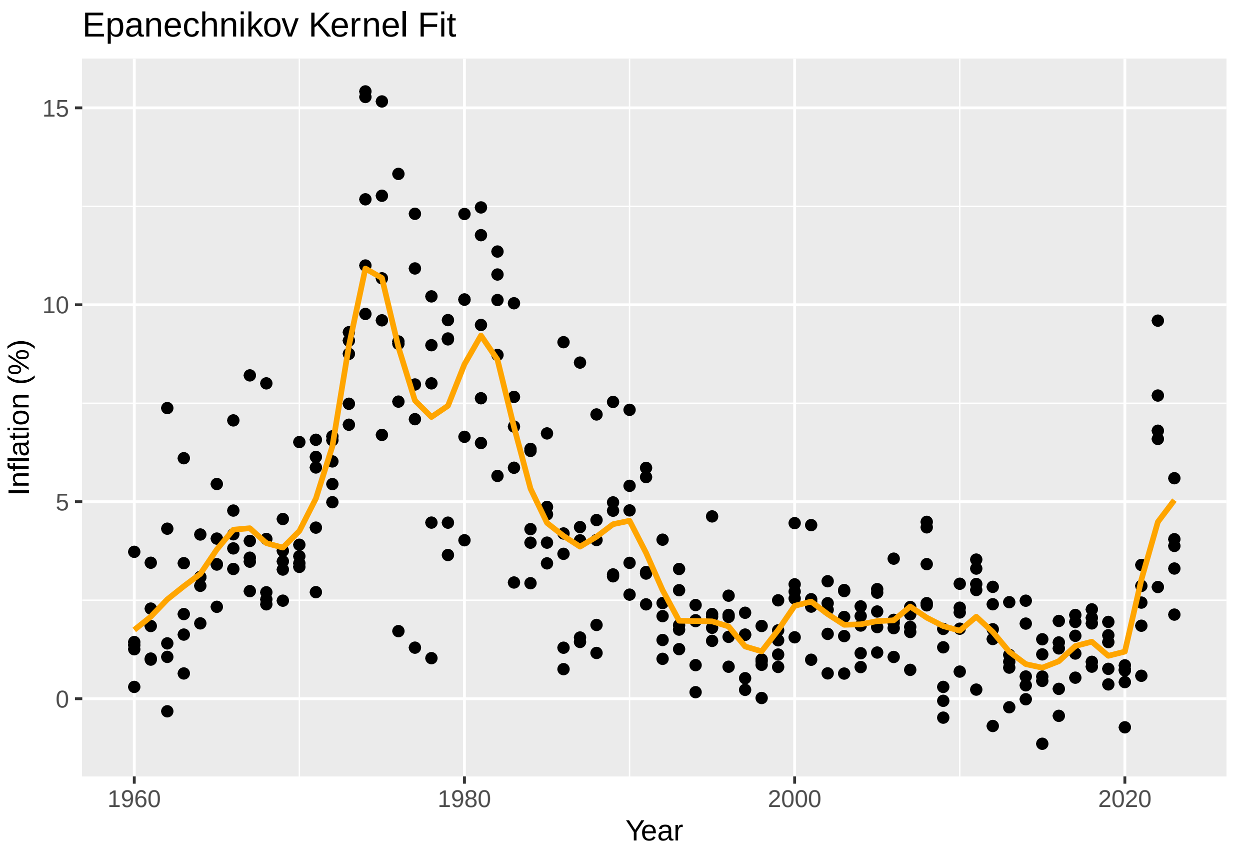
```
## Warning: package 'np' was built under R version 4.4.3
```

```
## Nonparametric Kernel Methods for Mixed Datatypes (version 0.60-18)
## [vignette("np_faq",package="np") provides answers to frequently asked questions]
## [vignette("np",package="np") an overview]
## [vignette("entropy_np",package="np") an overview of entropy-based methods]
```

```
epan_kernel <- npregbw(Inflation ~ Year, data = df01, regtype = "lc", ckertype = "epanechnikov")
```

```
## Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 /Multistart 1 of 1 |Multistar
```

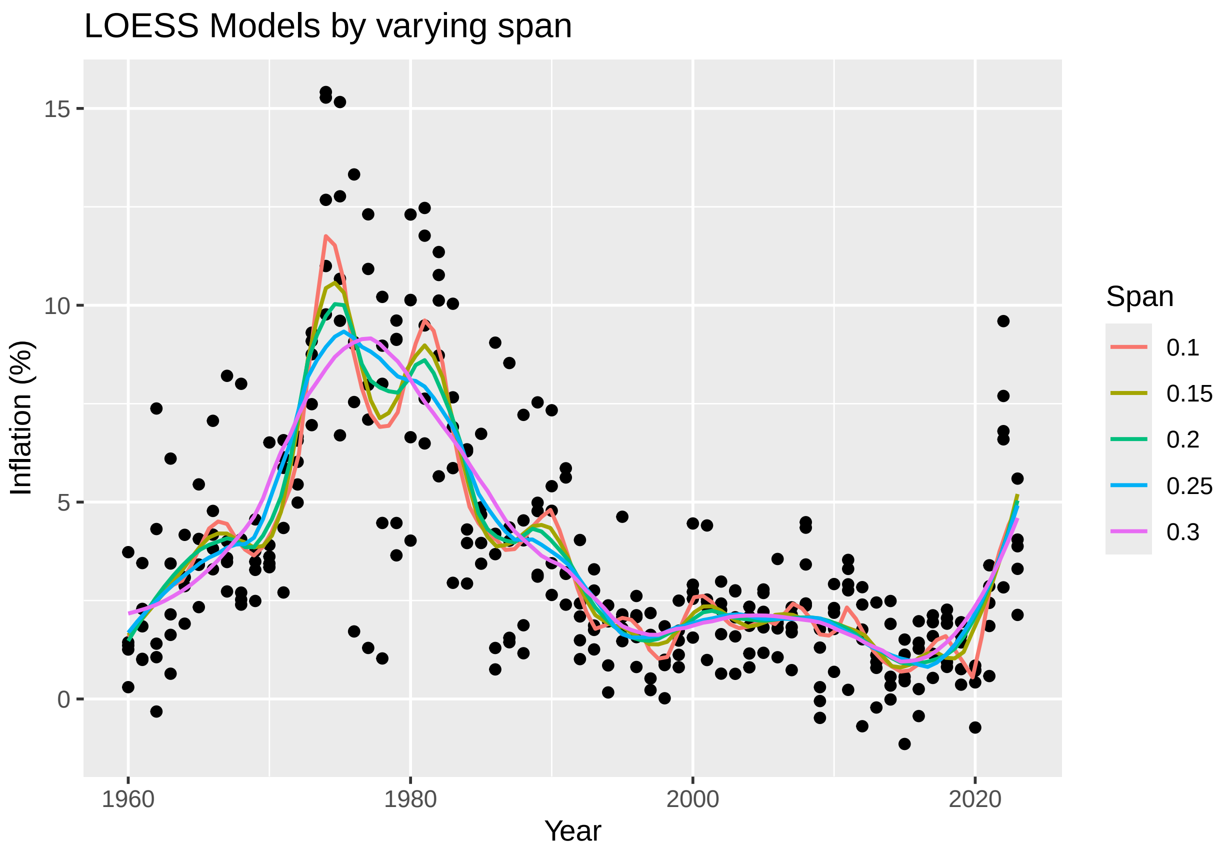
```r
fit <- npreg(epan_kernel)

fit.df <- data.frame(Year = df01$Year,
                     Inflation = df01$Inflation,
                     Fit = fitted(fit))

ggplot(fit.df, aes(x = Year))+
  geom_point(aes(y = Inflation))+
  geom_line(aes(y = Fit), color = "Orange", linewidth = 1)+
              labs(
                title = "Epanechnikov Kernel Fit",
                x = "Year",
                y = "Inflation (%)"
              )
```



2.3 Fitting LOESS.

When fitting localised regression, we are dealing with two parameters, degree in this instance is given as either 1 or 2 (linear or quadratic). Since we have determined that our data clearly follows a set of peaks and troughs we will set the degree as quadratic. The span parameter is our smoothing parameter in this case, it adjusts the size of the sliding window used to make predictions. Let's test a broad range of spans and plot the resulting fits.

```r
loess_predictions <- data.frame()

for (i in c(0.1, 0.15, 0.2, 0.25, 0.3)){
  loess_fit <- loess(Inflation ~ Year, data = df01, span = i, degree = 2)
  preds.loess <- predict(loess_fit, newdata = newdata, se = TRUE)
  temp <- data.frame(
    Year = newdata$Year,
    Prediction = preds.loess$fit,
    Span = factor(i))

  loess_predictions <- bind_rows(loess_predictions, temp)
}

ggplot(df01, aes(x = Year, y = Inflation)) +
  geom_point() +
  geom_line(
    data = loess_predictions,
    aes(x = Year, y = Prediction, color = Span),
    linewidth = 0.7
  ) +
  labs(
    title = "LOESS Models by varying span",
    x = "Year",
    y = "Inflation (%)",
    color = "Span"
  )
```



It is somewhat difficult to differentiate but the 0.2 span curve, feels like it best describes the change in inflation without over exaggerating or implausible fluctuations. For this reason we will save it as our final loess model.

```
loess_final <- loess(Inflation ~ Year, data = df01, span = 0.2, degree = 2)
```

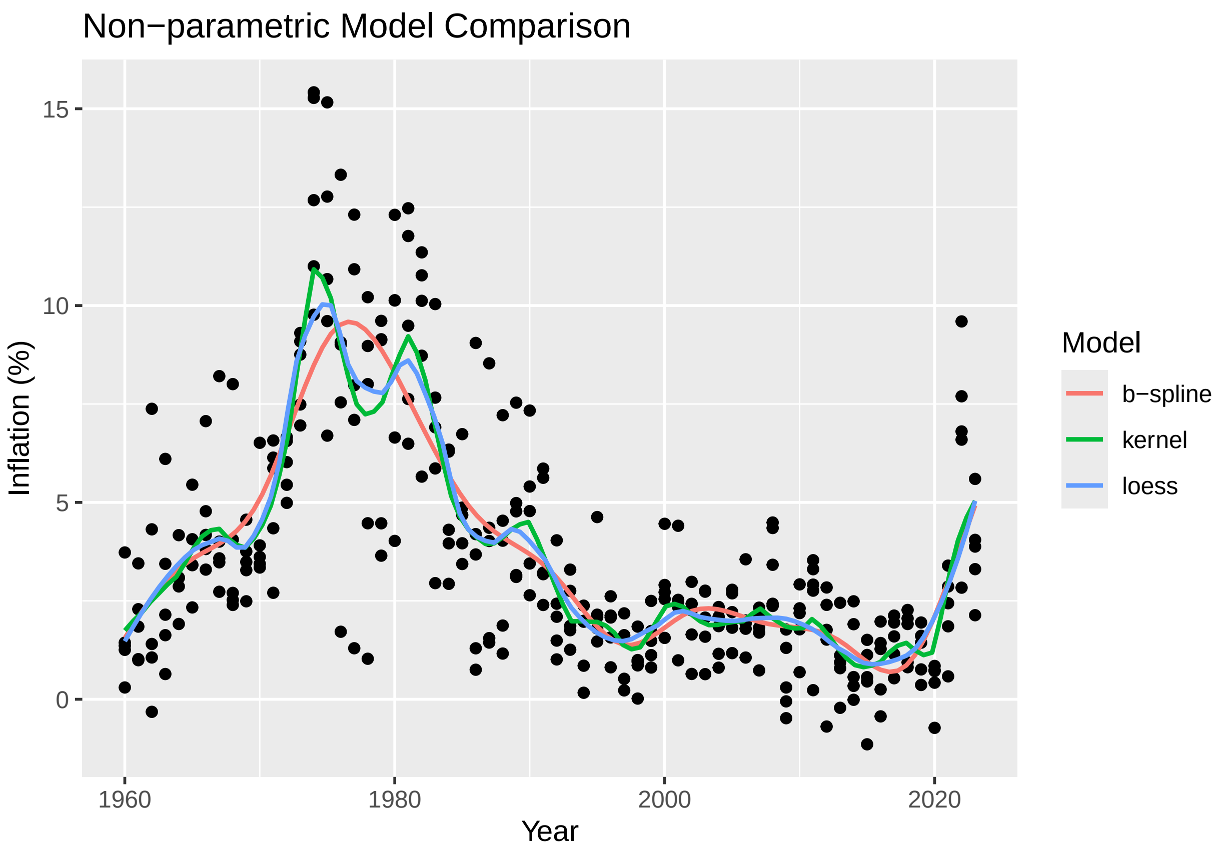
3 Comparing methods.

We are going to compare the fits from the different methods, with reference to features of the various fitted curves. In the next segments of code we will produce a single plot using ggplot() that shows all the fitted curves; polynomial, kernel, spline and loess.

```
loess.fp <- predict(loess_final, newdata = newdata)
bspline.fp <- predict(final_blspine, newdata = newdata)
epan_kernel <- npregbw(Inflation ~ Year, data = df01, regtype = "lc", ckertype = "epanechnikov")
```

```
## Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 |Multistart 1 of 1 /Multistart 1 of 1 |Multistar
```

```
fit <- npreg(epan_kernel)
kernel.fp <- predict(fit, newdata = newdata)

loess.df <- data.frame(Year = newdata$Year,
    Prediction = loess.fp,
    Model = "loess")

bspline.df <- data.frame(Year = newdata$Year,
    Prediction = bspline.fp,
    Model = "b-spline")

kernel.df <- data.frame(Year = newdata$Year,
    Prediction = kernel.fp,
    Model = "kernel")

final_predictions_all_models <- bind_rows(loess.df, bspline.df, kernel.df)

ggplot(df01, aes(x = Year, y = Inflation)) +
  geom_point() +
  geom_line(data = final_predictions_all_models,
            aes(x = Year, y = Prediction, color = Model),
            linewidth = 0.8) +
  labs(title = "Non-parametric Model Comparison",
       y = "Inflation (%)",
       color = "Model")
```



All three models agree on the major macro trends the sharp rise in inflation in the 1970s - 1980's, its decline through the 1990s, and the resurgence in the early 2020s. This indicates robustness across models in capturing non linear structures. In regards to the research question, we found that all three models provide a substantial representation for the change in inflation over time.

The B-spline model provided the smoothest overall fit, however, when we increased the degrees of freedom to better capture extreme values in the data, it began to produce unrealistic fluctuations, likely due to overfitting. This is because as the df increases the number of knots increase and the spline fits to more local variation.

We used the np package's bandwidth selection to fit the Epanechnikov kernel regression. The method minimises CV prediction error, balancing tradeoff variance bias. This approach allows for a smooth, flexible fit without needing to manually specify tuning parameters such as polynomial degree or knot locations, as required by spline methods. Subjectively, the kernel was our worst performing model. Slight edge effects were observed near the boundaries of the data, a known limitation of kernel methods. It also suffered from jaggedness and clear overfitting.

Lastly, we applied LOESS which we found to be our most flexible approach for fitting inflation data. By fitting simple models within a moving window, LOESS adapts well to nonlinear patterns without assuming a global form. This made it effective in highlighting the fluctuations in inflation in a way which was less drastic than our kernel estimator. However, the method is sensitive to the choice of span, with a smaller span, we observed excessive variability, while a larger span smoothed over meaningful local changes. Despite this tradeoff, we feel that LOESS offered a well balanced and malleable option for the dataset, making it one of the stronger methods in our comparison.

When looking to describe trends rather than extrapolate (our goal in regards to the research question), it should be noted that all 3 non parametric models met this goal when compared to standard polynomial regression.