Distributed Lag Non-linear Models (DLNM)

Methodology and Application to Time Series Data Analysis

Vincent Arumadri

Department of Public Health

Erasmus MC

2025

Key Definitions

- Non-linear data structure: data where there is no linear relationship between a dependent (outcome/response) and an independent (exposure/predictor) variable
- Time series data: a sequence of data points collected over an interval of time e.g daily rainfall measurements, weekly sales
- Lag: time difference between two observations in a sequence, or the number of steps back in time (delay) a past observation is from the current time

Day	Observed Value	Lag-1	Lag-2
1	10		
2	20	10	
3	30	20	10
4	40	30	20
5	50	40	30

Key Definitions

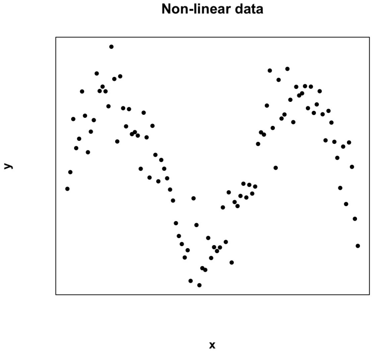
- Autoregressive model: specifies that the response variable depends linearly on its own previous (lagged) value(s)

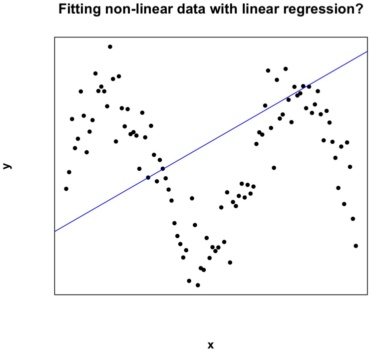
$$\underbrace{y_t}_{\text{Response variable}} = \beta_0 + \underbrace{\beta_2 y_{t-1}}_{\text{Lagged response variable as predictor}} + \epsilon$$

- Distributed lag model: specifies that the response variable depends linearly on the current and lagged value(s) of predictor variables and no lagged response variables

$$y_t = \beta_0 + \underbrace{\beta_1 x_t}_{\text{Current predictor}} + \underbrace{\beta_2 x_{t-1}}_{\text{Lagged predictor}} + \epsilon$$

Non-linearity





- Single linear function under-fits non-linear data

Polynomial Regression

- Extend linear models to flexibly model non-linear relationships by including polynomial terms (x^2, x^3,...,x^d) for a predictor
- Instead of a linear model:

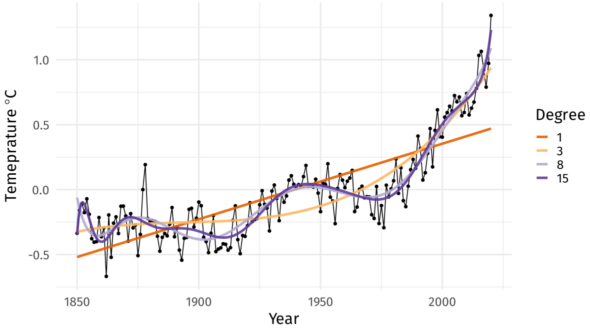
$$\underbrace{\overbrace{y}^{\text{response variable}} = \underbrace{\beta_0}_{\text{intercept}} + \overbrace{\beta_1 x}^{\text{predictor}} + \underbrace{\epsilon}_{\text{error/random term}}}$$

- Create (extra) predictors by raising each of the original predictors to a power

$$y_i = \beta_0 + \overbrace{\beta_1 x}^{\text{Linear term (degree 1)}} + \underbrace{\beta_2 x^2}_{\text{Degree 2 polynomial}} + \underbrace{\beta_3 x^3}_{\text{Degree 3 polynomial}} + \ldots + \overbrace{\beta_d x^d}^{\text{Degree d polynomial}} + \epsilon$$

Polynomial Regression

- Unknown coefficients β_1, $\beta_2 \ldots \beta_d$ can easily be estimated using least squares because this is just a linear model with non-linear function of predictors (x, x^2, x^3,...,x^d)
- Unusual to use d greater than 3 or 4, overfitting and wiggly



Function of predictors (basis function)

- Known family of functions or transformations e.g polynomial functions, threshold functions, regression splines etc applied to a predictor X to generate basis variables: $b_1(X)$, $b_2(X),\ldots,b_K(X)$.

$$y = \beta_0 + \beta_1 b_1(x) + \beta_2 b_2(x) + \ldots + \beta_k b_k(x) + \epsilon \quad (1)$$

- For polynomial regression, the basis function takes the form:

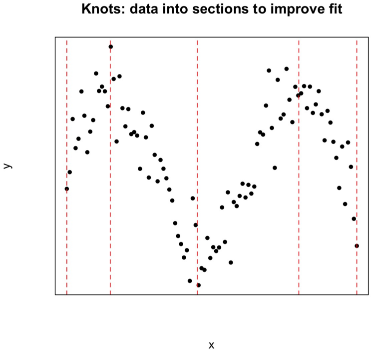
$$\underbrace{b_j(x)}_{\text{polynomial function of degree j}} = \underbrace{x^j}_{\text{Raise predictor x to degree j}} \quad (2)$$

- Substituting equation (2) in (1)

$$y = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \ldots + \beta_k x^k + \epsilon$$

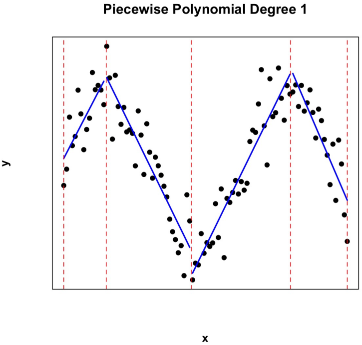
Modelling Non-linear relationships: Knots

- A first step is to divide the data into sections and generate individual regressions in each
- These divisions inside the predictor, happen at positions called knots, ξ
- Coefficients change at knots, more knots in places where we feel the function might vary most rapidly



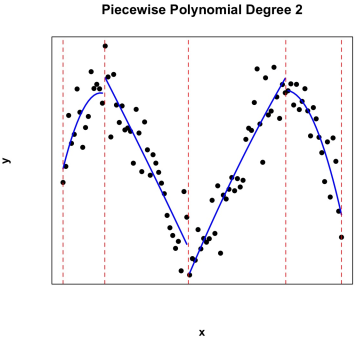
Piecewise Polynomial Regression

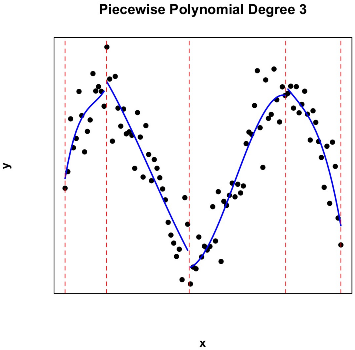
$$y_i = \begin{cases} \beta_{01} + \beta_{11}x + \epsilon & \text{if } x < \xi_1, \\ \beta_{02} + \beta_{12}x + \epsilon & \text{if } \xi_1 \geq x < \xi_2, \\ \beta_{03} + \beta_{13}x + \epsilon & \text{if } \xi_2 \geq x < \xi_3, \\ \beta_{04} + \beta_{14}x + \epsilon & \text{if } \xi_3 \geq x < \xi_4. \end{cases}$$



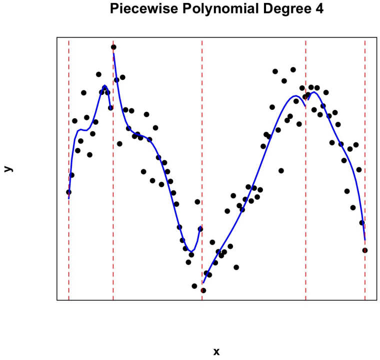
- Applying degree 1 polynomial basis functions to predictor variables X to generate basis variables of predictor X, 'piecewise'

Piecewise Polynomial Regression





Piecewise Polynomial Regression



- A major pathology to notice in all of the above examples is the discontinuity at the knots (unconstrained polynomials)
- The functions are continuous
- Unconstrained here means no restrictions of continuity imposed at knots

Regression Splines

- We want continuity at the knots, and smoothness of the overall curve, we impose continuity constraints
- Regression spline with knots at ξ_k, k=1,...,K is a piecewise polynomial continuous at each knot
- Linear spline is represented with a basis for a linear polynomial x and one truncated power basis function per knot

$$y = \beta_0 + \overbrace{\beta_1 x}^{\text{Linear polynomial term (degree 1)}} + \sum_{k=1}^{K} \underbrace{\beta_k (x - \xi_k)_+}_{\text{Truncated power basis}} + \epsilon$$

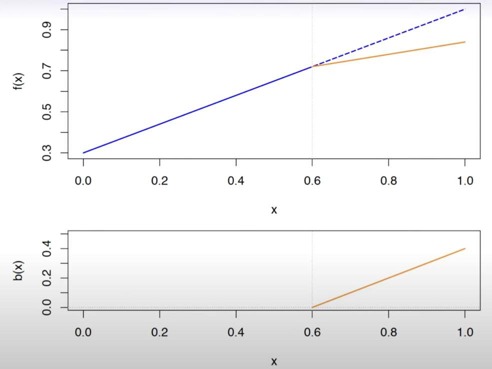
- Ensures continuity at knots

Truncated Power Basis Function

- Truncated power basis function is zero for $x \leq \xi_k$ and nonzero only for $x > \xi_k$

$$(x-\xi)_+ = \begin{cases} (x-\xi), & \text{if } x > \xi \\ 0, & \text{otherwise} \end{cases}$$

- Linear spline representation using truncated power basis function at knot, $\xi = 0.6$

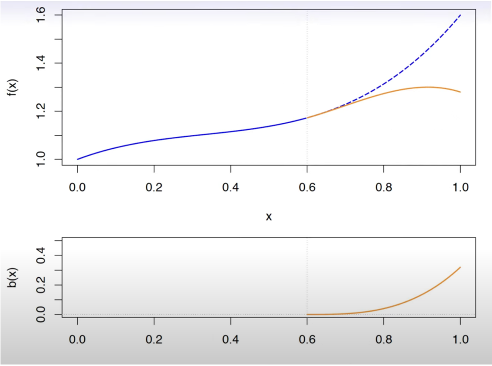


Truncated Power Basis Function

- Truncated power basis function is zero for $x \leq \xi_k$ and nonzero only for $x > \xi_k$

$$(x-\xi)_+^3 = \begin{cases} (x-\xi)^3, & \text{if } x > \xi \\ 0, & \text{otherwise} \end{cases}$$

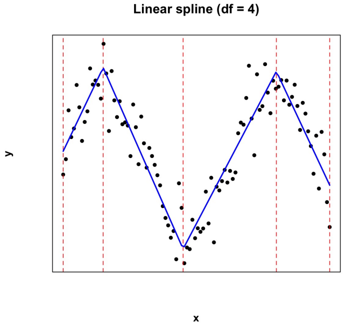
- Cubic spline representation using truncated power basis function at knot, $\xi = 0.6$



C^0 Continuity

- Function values must match at each knot, ξ, meaning there are no gaps or jumps e.g for a linear spline

$$y = \beta_0 + \beta_1 x + \underbrace{\beta_2 (x - \xi)_+}_{\text{Ensures continuity at knots}} + \epsilon$$

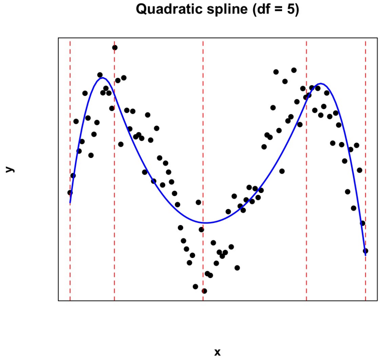


$$y = \begin{cases} \beta_0 + \beta_1 x + \epsilon & \text{if } x < \xi_1, \\ .. + \beta_2 (x - \xi_1) + \epsilon & \text{if } \xi_1 \geq x < \xi_2, \\ ... + \beta_3 (x - \xi_2) + \epsilon & \text{if } \xi_2 \geq x < \xi_3, \\ .... + \beta_4 (x - \xi_3) + \epsilon & \text{if } \xi_3 \geq x < \xi_4. \end{cases}$$

- $\xi_1 \geq x < \xi_2$, first truncated basis activated, adding a slope change

C^1 Continuity

- First derivative must be continuous at each knot, ξ, ensuring no abrupt changes in curvature/direction i.e no sharp ends e.g for a quadratic spline

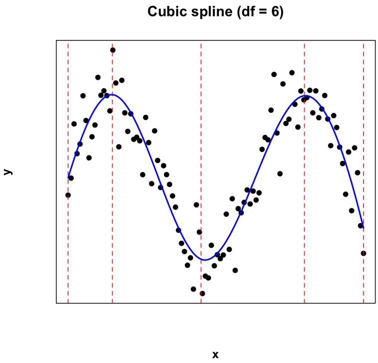


$$(x-\xi)_+^2 = \begin{cases} (x-\xi)^2, & x > \xi \\ 0, & x \leq \xi \end{cases}$$

$$\frac{d}{dx}(x-\xi)_+^2 = \begin{cases} 2(x-\xi), & x > \xi \\ 0, & x \leq \xi \end{cases}$$

C^2 Continuity

- Second derivative must be continuous at each knot, ξ, ensuring smooth curvature e.g for a cubic spline



$$(x-\xi)_+^3 = \begin{cases} (x-\xi)^3, & x > \xi \\ 0, & x \le \xi \end{cases}$$

$$\frac{d}{dx}(x-\xi)_+^3 = \begin{cases} 3(x-\xi)^2, & x > \xi \\ 0, & x \le \xi \end{cases}$$

$$\frac{d^2}{dx^2}(x-\xi)_+^3 = \begin{cases} 6(x-\xi), & x > \xi \\ 0, & x \le \xi \end{cases}$$

- The cubic spline has a more smooth curvature and fits the data better than the quadratic spline

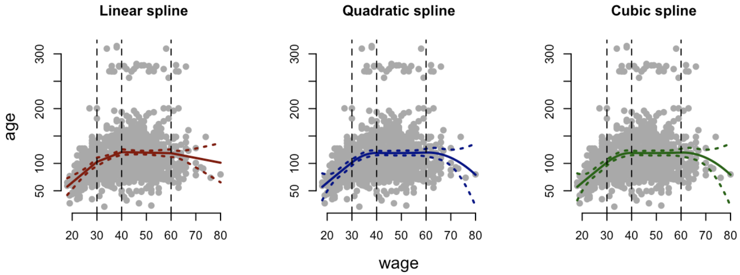
Comparison of Splines

Spline Type	Function Continuous? (C^0)	1st Derivative Continuous? (C^1)	2nd Derivative Continuous? (C^2)	Use case
Linear	**Yes (No gaps/jumps)**	No (b)	No (b)	Simple trends
Quadratic	Yes	**Yes (No sharp ends)**	No (a)	Some curvature, but limited smoothness
Cubic	Yes	Yes	**Yes (No abrupt changes in curvature)**	**Smooth curves, most common choice**

- Cubic splines, seem to be the best tool we have so far, right? Well, there is always room for improvement

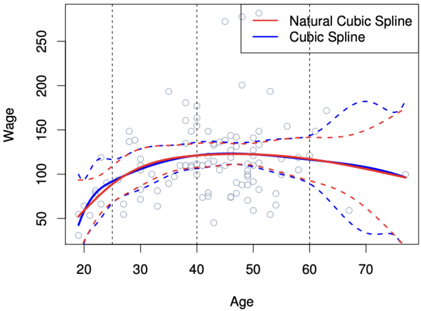
Comparison of Splines

- B-Splines can have high variance at the outer range of the predictors—that is, when X takes on either a very small or very large value



Natural (Restricted) Cubic Spline

- Natural cubic spline: spline with additional boundary constraints, enforcing linearity beyond boundary knots
- Means that natural cubic splines generally produce more stable estimates at the boundaries (narrower confidence intervals) than cubic splines



Natural (Restricted) Cubic Spline

- Ensures linearity beyond the boundary by enforcing the second derivative of cubic splines to be zero at the boundary knots

$$\left.\frac{d^2 f}{dx^2}\right|_{x<\xi_1} = 0, \quad \left.\frac{d^2 f}{dx^2}\right|_{x_n>\xi_k} = 0$$

- Use lower-degree (3) polynomials making them more flexible (less wiggles and overfitting) and therefore computationally efficient
- Splines provide a transparent and interpretable fit (linear in coefficients with non-linear function of predictors)

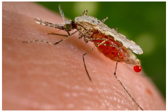
- **How are they used in DLNMs?**

DLNM: Conceptual Model

Heavy rainfall
(Exposure)

Leads

Exposure-response



Malaria cases
(Response)

Heavy rainfall
(Exposure)

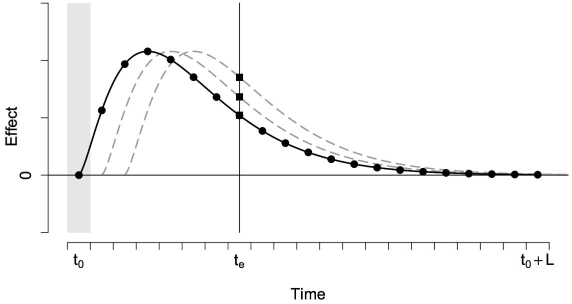
Delayed effect?

Exposure-lag-response

Malaria cases
(Response)

DLNM: Conceptual Model

- DLNMs capture a detailed representation of the time-course of the exposure-**lag**-response relationship
- Risk associated with individual exposure events at each lag assigned a weight that contributes to overall cumulative risk



- **Statistical issue is to model this risk!**

Exposure-lag-response associations

- The risk is represented by a function $s(x, t)$ defined in terms of both **intensity** and **timing** of a series of **past exposures**:
  - an **exposure-response** function $f(x)$ for **exposure** x
  - a **lag-response** function $w(\ell)$ for **lag** ℓ
- Generating a **bi-dimensional exposure-lag-response** function $f \cdot w(\boldsymbol{x}, \ell)$:

$$s(x, t) = \int_{\ell_0}^{L} f(x_{t-\ell}) \cdot w(\ell)\, d\ell$$

- Approximation obtained through a **discretization of the lag period into equally spaced time units**

$$s(x_{t-\ell_0}, \dots, x_{t-L}) \approx \sum_{\ell=\ell_0}^{L} f(x_{t-\ell}) \cdot w(\ell)$$

Basic Model

- A general model representation to describe the time series of outcomes Y_t with $t = 1, \ldots, n$ is given by

Link function

Smoothed predictor

$$g(\mu_t) = \alpha + \sum_{j=1}^{J} s_j(x_{tj}; \beta_j) + \sum_{k=1}^{K} \gamma_k u_{tk}$$

Other predictors with linear effects

- where $\mu \equiv E(Y)$, and Y is assumed to arise from a distribution belonging to the exponential family
- x_{tj} is the transformed exposure at time t through basis function j
- β_j is unknown coefficient of x_{tj} to be estimated

Distributed lag models (DLMs)

- Response Y at a given time t, (Y_t), may be explained in terms of **past exposures** $x_{t-\ell}$ with ℓ for lags $\ell = \ell_0, \dots, \ell, \dots, L$:

$$q_{x_t} = [x_{t-\ell_0}, \dots, x_{t-\ell}, \dots, x_{t-L}]$$

- Assuming a **linear exposure-response**, hence **DLM**, we can write:

$$s(x, t; \beta) = \sum_{\ell=\ell_0}^{L} x_{t-\ell} \cdot w(\ell) = q_{x_t} C \beta = w_{x_t} \beta$$

- C is obtained by applying a **specific basis transformation** to the lag vector $\ell = [\ell_0, \dots, \ell, \dots, L]$
- w_{xt} is the vectorial representation of integral of $x \cdot w\ell$ over the interval $[\ell_0, L]$ with parameters β.

Distributed lag non-linear models (DLNMs)

- Assuming a **non-linear exposure-response relationship**, hence **DLNM**, the function becomes;

$$s(x_{t-\ell_0}, \ldots, x_{t-L}) \approx \sum_{\ell=\ell_0}^{L} f(x_{t-\ell}) \cdot w(\ell)$$

- **Disadvantage** imposed by the **assumption of independence** of the exposure-response and the lag-response function

$$s(x_{t-\ell_0}, \ldots, x_{t-L}) \approx \sum_{\ell=\ell_0}^{L} f \cdot w(x_{t-\ell}, \ell)$$

- We can model the exposure-response association and the lag-response association **simultaneously**

Distributed lag non-linear models (DLNMs)

- R_{x_t} is obtained by applying a **second basis transformation** to q_{x_t}
- Then we define a **special tensor product**:

$$A_{x_t} = (1_{v_\ell} \otimes R_{x_t}) \odot (C \otimes 1_{v_x})$$

which forms the **crossbasis**:

$$s(q_{x_t}; \beta) = (1_{v_x.v_\ell} A_{x_t})\beta = w_{x_t}\beta$$

- **The problem reduces to choosing a basis for each** q_{x_t} **and** ℓ, defining **exposure-response** and **lag-response** functions, respectively

Basis for exposure space

- Given, Q_x (timeseries of exposure X) and assuming a maximum lag of 2, we can compute, Q_{xt} (vector of lagged exposure histories of X)
- Applying **linear transformation** to Q_{xt} to get R_{xt} (lagged occurrences of each of the basis variables of X)

t	x
1	10
2	20
3	30
4	40
5	50

$\Rightarrow$

t	**lag 0**	**lag 1**	**lag 2**
1	10	NA	NA
2	20	10	NA
3	30	20	10
4	40	30	20
5	50	40	30

$$R_{xt} \Rightarrow \begin{bmatrix} 10 & NA & NA \\ 20 & 10 & NA \\ 30 & 20 & 10 \\ 40 & 30 & 20 \\ 50 & 40 & 30 \end{bmatrix}$$

Basis for lag Space

- Applying **polynomial transformation** of degree 2 to the lag vector, ℓ(0,1,2)
- First step is to scale the lag vector by dividing by the maximum lag:

$$(0,1,2)/2 \Rightarrow (0,0.5,1)$$

- Applying polynomial transformation to the lag vector ℓ, to obtain C (basis variables for each lag for degrees d = 0,1,2)

x^d	x^0	x^1	x^2
lag 0[0]	1	0	0
lag 1[0.5]	1	0.5	0.25
lag 2[1]	1	1	1

$$C \Rightarrow \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{bmatrix}$$

Special Tensor Product

Kronecker product (matrix direct product)

$$A_{xt} = (1_{vl} \otimes R_{xt}) \odot (C \otimes 1_{vx})$$

Hadamard product (element-wise product)

- 1_{vx}: Vector of 1's of dimensional length of exposure

t	Q_x
1	10
2	20
3	30
4	40
5	50

$$1_{vl} \Rightarrow \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

- 1_{vl}: Vector of 1's of dimensional length of lag vector

$$c(0,1,2), 1_{vl} \Rightarrow [1,1,1]$$

Special Tensor Product

- The first part $1_{vl} \otimes R_{xt}$ makes copies of each exposure basis per lag

$$
[1, 1, 1] \otimes \begin{bmatrix} 10 & NA & NA \\ 20 & 10 & NA \\ 30 & 20 & 10 \\ 40 & 30 & 20 \\ 50 & 40 & 30 \end{bmatrix} \Rightarrow \begin{bmatrix} 1 \cdot R_{xt}[i] \\ 1 \cdot R_{xt}[i] \\ 1 \cdot R_{xt}[i] \end{bmatrix} = \left[\begin{array}{ccc} 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \\ 40 & 40 & 40 \\ 50 & 50 & 50 \\ \hline NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \\ 40 & 40 & 40 \\ \hline NA & NA & NA \\ NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \end{array}\right]
$$

Special Tensor Product

- The second part $C \otimes 1_{vx}$ makes copies of each lag basis per exposure

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1 & 1 \end{bmatrix} \otimes \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \Rightarrow \begin{bmatrix} C[i] \cdot 1 \\ C[i] \cdot 1 \\ C[i] \cdot 1 \\ C[i] \cdot 1 \\ C[i] \cdot 1 \end{bmatrix} = \left[\begin{array}{ccc} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ \hdashline 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ \hdashline 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{array}\right]$$

Special Tensor Product

- Element-wise multiplication of the first part and second part $(1_{vl} \otimes R_{xt}) \odot (C \otimes 1_{vx})$ results into;

$$\begin{bmatrix} 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \\ 40 & 40 & 40 \\ 50 & 50 & 50 \\ \hline NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \\ 40 & 40 & 40 \\ \hline NA & NA & NA \\ NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \end{bmatrix} \odot \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \\ \hline 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ 1 & 0.5 & 0.25 \\ \hline 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \Rightarrow A_{xt} = \begin{bmatrix} 10 & 0 & 0 \\ 20 & 0 & 0 \\ 30 & 0 & 0 \\ 40 & 0 & 0 \\ 50 & 0 & 0 \\ \hline NA & NA & NA \\ 10 & 5 & 2.5 \\ 20 & 10 & 5 \\ 30 & 15 & 7.5 \\ 40 & 20 & 10 \\ \hline NA & NA & NA \\ NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \end{bmatrix}$$

Special Tensor Product

- From Gasparrini et al. 2010 "... array A_{xt} than re-arranged **summing along the third dimension of lags** to obtain the final matrix of cross-basis functions, w_{xt}."

$$\Rightarrow \begin{bmatrix} 10 & 0 & 0 \\ 20 & 0 & 0 \\ 30 & 0 & 0 \\ 40 & 0 & 0 \\ 50 & 0 & 0 \end{bmatrix} \oplus \begin{bmatrix} NA & NA & NA \\ 10 & 5 & 2.5 \\ 20 & 10 & 5 \\ 30 & 15 & 7.5 \\ 40 & 20 & 10 \end{bmatrix} \oplus \begin{bmatrix} NA & NA & NA \\ NA & NA & NA \\ 10 & 10 & 10 \\ 20 & 20 & 20 \\ 30 & 30 & 30 \end{bmatrix}$$

- E.g for [1,1] 10 + NA + NA = NA; [5,1] 50 + 40 + 30 = 120

$$w_{xt} = \begin{bmatrix} NA & NA & NA \\ NA & NA & NA \\ 60 & 20 & 15 \\ 90 & 35 & 27.5 \\ 120 & 50 & 40 \end{bmatrix} \Rightarrow w_{xt}\beta$$

Cross-basis matrix

```
# Load package
library(dlnm)
```

```
## This is dlnm 2.4.7. For details: help(dlnm) and vignette('dlnmOverview').
```

```
# data
x <- data.frame(
  time = 1:5,
  value = c(10, 20, 30, 40, 50)
)
x
```

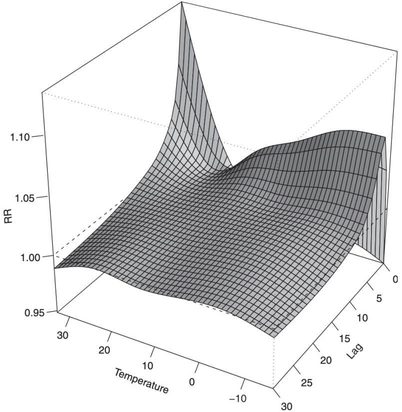
```
##   time value
## 1    1    10
## 2    2    20
## 3    3    30
## 4    4    40
## 5    5    50
```

```
# cross-basis
y <- crossbasis(x$value, lag = 2, argvar = list(fun="lin"),
                arglag = list(fun="poly", degree=2))
head(y,5)
```

```
##      v1.l1 v1.l2 v1.l3
## [1,]    NA    NA    NA
## [2,]    NA    NA    NA
## [3,]    60    20  15.0
## [4,]    90    35  27.5
## [5,]   120    50  40.0
```

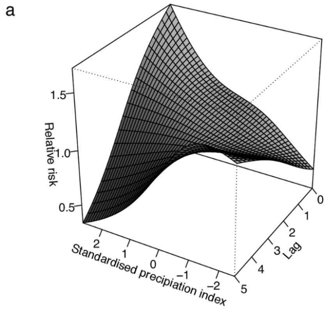
Example 1

- Effect of temperature on overall mortality in New York City, during the period 1987–2000

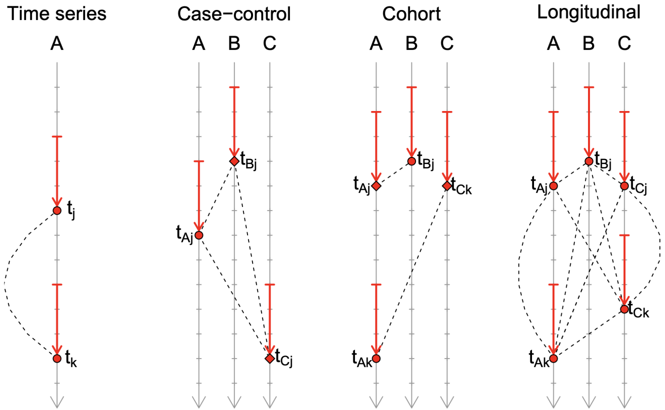


Example 2

- Effect of rainfall and temperature on dengue risk in Barbados during the period 1999-2016



Alternative Study Designs



References

- A. Gasparrini, Armstrong, and Kenward 2010
- Antonio Gasparrini 2011
- Gareth James • Daniela Witten • Trevor Hastie and Robert Tibshirani 2013
- Antonio Gasparrini and Leone 2014
- Aßenmacher 2016
- Lowe et al. 2018

Thank You!