Reverse engineering Atmospheric Dust Content from Engine Samples

ITT 20: Rolls-Royce

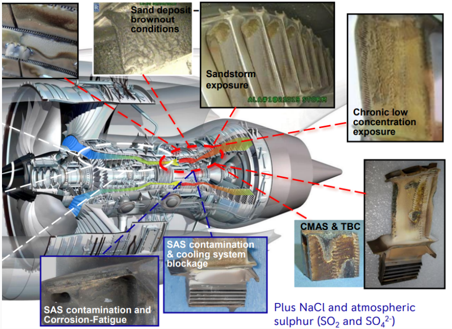
Amin Sabir **Bill Nunn** **Daniel Hajnal** **Matt Evans**

Supervised by: Matt Nunes, Sergey Dolgov, Theresa Smith

University of Bath

Problem overview

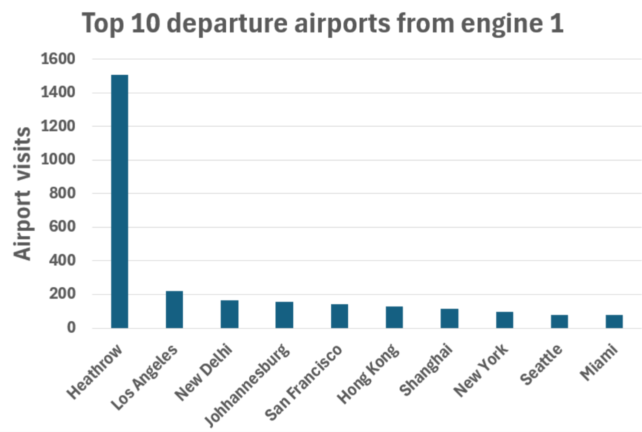
- As a jet engine is used it will accumulate dust internally
  - **Most** dust from take-off
  - **Mixed** in the engine
- The type of dust varies **geographically**
  - Some types are **more damaging** than others


Sand deposit – brownout conditions
Sandstorm exposure
Chronic low concentration exposure
CMAS & TBC
SAS contamination & cooling system blockage
SAS contamination and Corrosion-Fatigue
Plus NaCl and atmospheric sulphur (SO_2 and SO_4^{2-})

- Can we **infer** what type of dust is present at **each** airport?

Dataset

- **Synthetic** proportions of **5 types** of accumulated dust (oxides) from each engine
- Data from 20 engines over 58k **real** flight paths from 298 airports
- **Synthetic ground truth** for dust-concentration per airport



Two approaches

1. **Inverse problems** on this dataset
2. **Bayesian approach** on some toy examples

Inverse problem approach

Linear system:

$$A\boldsymbol{x} = \boldsymbol{y}$$

- Final vector $\boldsymbol{y}$ - the 5 different dust **concentrations** in the engines
- Dust-concentration vector $\boldsymbol{x}$ of all the airports $[(N_{\text{airports}} \cdot 5) \times 1]$
- **Under-determined** linear system
- Specify a suitable **forward model** of A - airport to engine dust-concentrations

Forward model for A

- Assume each airport has a time-independent concentration vector, $\mathbf{x}_i$
- Final concentration vector $\mathbf{y}$ in a given engine

Break down the $\mathbf{y}$ into a sum over the flights:

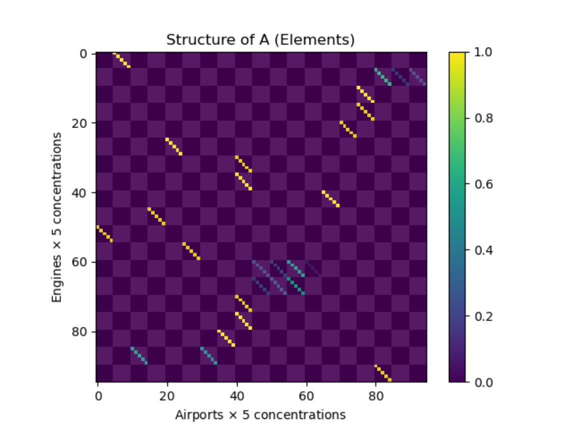
$$\mathbf{y} = \sum_i \frac{1}{(m_{i,in} + m_{i,out})} \sum_{flights;i} m_{i,in}\mathbf{x}_{i,in} + m_{i,out}\mathbf{x}_{i,out}$$

Now assume that the mass (m) is **known** and is the **same** at each airport:

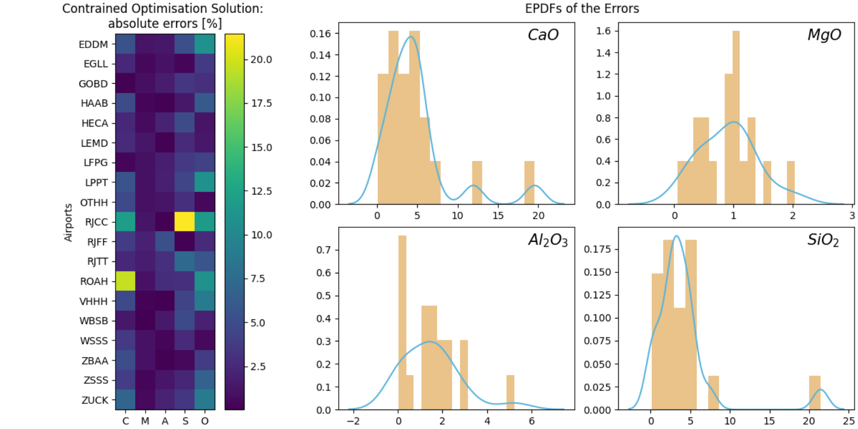
$$\mathbf{y} = \underbrace{\frac{1}{n_{flights}} \sum_{flights;i} \mathbf{x}_i}_{Ax}$$

Least-squares method

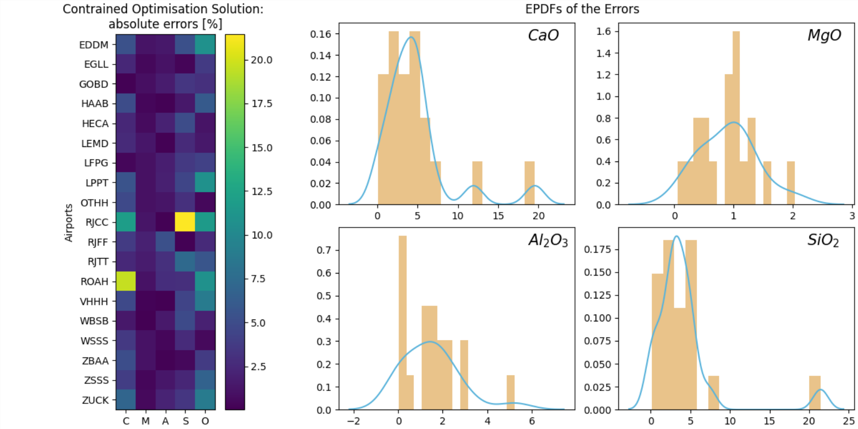
- Reduce A to a **square** matrix (19 airports visited $>$ 350 times)
- Take the pseudo-inverse of A
- Constrain predicted results to be positive and sum to 1 (full dust concentration)



Preliminary results



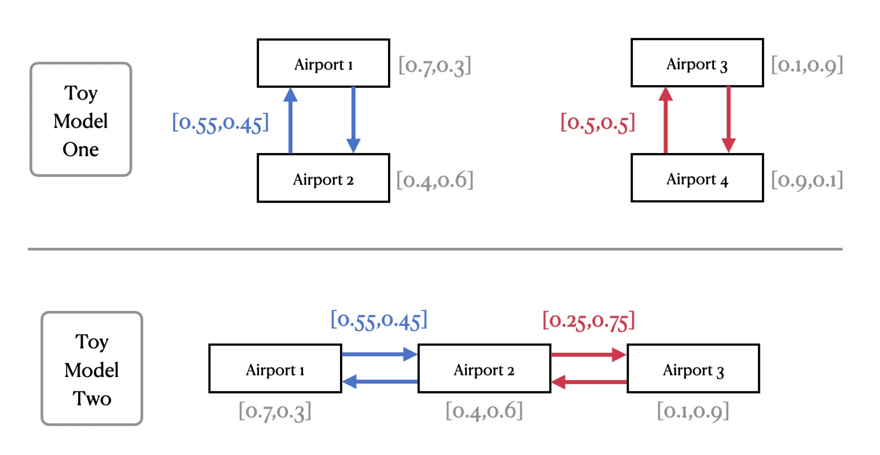
Preliminary results



- Extensions: time-dependence *x* by weighing recent flights **more** than earlier - **geometric decay**

Exact inference toy examples

- We assume each airport has two types of dust, two engines are tracked, with flight patterns shown in blue and red



A Bayesian model for these toy examples

- We sample the posterior distribution

  $$\mathbb{P}(x_j|y_i) \propto \mathbb{P}(y_i|x_j) \times \mathbb{P}(x_j)$$

  using Markov chain Monte Carlo

A Bayesian model for these toy examples

- We sample the posterior distribution

$$\mathbb{P}(\boldsymbol{x}_j|\boldsymbol{y}_i) \propto \mathbb{P}(\boldsymbol{y}_i|\boldsymbol{x}_j) \times \mathbb{P}(\boldsymbol{x}_j)$$

  using Markov chain Monte Carlo

- The forward model (a.k.a likelihood) for the proportion of each dust type in the first engine is

$$\boldsymbol{y}_1 \sim \text{Normal}\left(\frac{1}{2}(\boldsymbol{x}_1 + \boldsymbol{x}_2), \sigma^2 I\right)$$

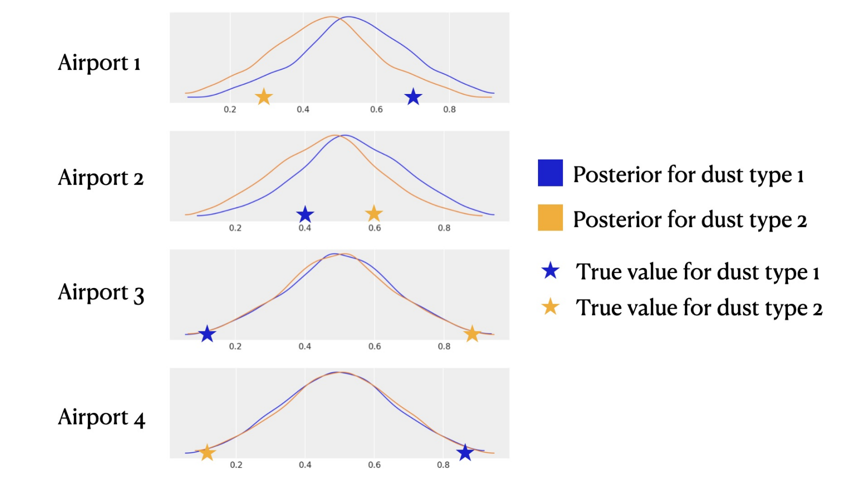
- The prior beliefs for the proportion of each dust type at airport j are

$$\boldsymbol{x}_j \sim \text{Dirichlet}(\boldsymbol{\alpha}_j)$$

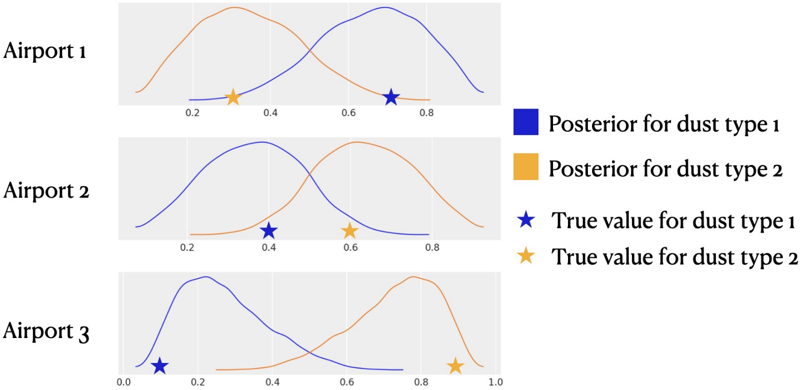
$$\boldsymbol{\alpha}_j \sim \text{Uniform}([10, 100]^2)$$

- The posterior sampling means approximates the pseudo inverse solution discussed by Amin

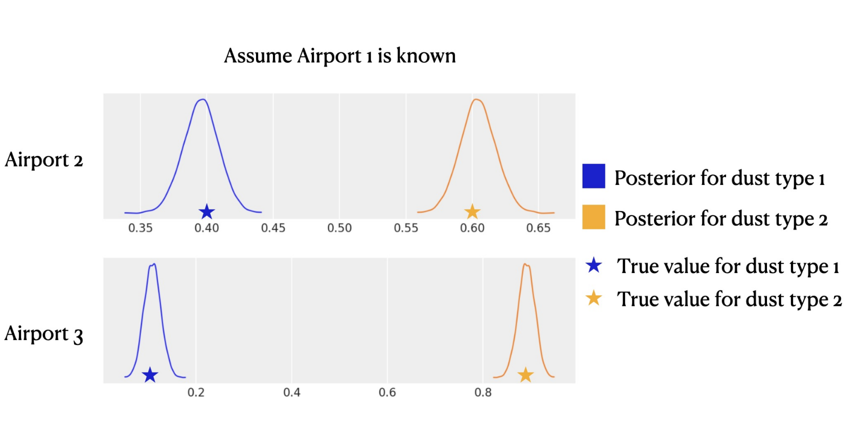
Results: two pairs of airports



Results: pairs of airports one shared



Results: pairs of airports one shared



Much more certain estimates, centred around the ‘true values’

Extending this Bayesian model

- We can apply a similar Bayesian methodology to the full data set, yielding a strict extension of the pseudo-inverse method
- The Bayesian methods are **much** more computationally costly

Extending this Bayesian model

- We can apply a similar Bayesian methodology to the full data set, yielding a strict extension of the pseudo-inverse method
- The Bayesian methods are **much** more computationally costly
- There are other natural ways to incorporate additional information in the Bayesian model
    - Pooling of the Dirichlet parameters based on **geography**
    - Hard coding other knowledge-based constraints. For instance we can encode knowledge that there is no dust of a certain type at a particular airport in the Dirichlet parameter priors

Conclusions and next steps

Conclusions

- **Inverse problem method**: reasonable approach for finding the mean dust concentrations per airport
- **Bayesian approach**: strictly extends the inverse problem method. Understanding uncertainty via posterior sampling is essential

Conclusions and next steps

Conclusions

- **Inverse problem method**: reasonable approach for finding the mean dust concentrations per airport
- **Bayesian approach**: strictly extends the inverse problem method. Understanding uncertainty via posterior sampling is essential

Next steps

- Extend the forward model to include time dependence with seasonality
- Use the CAMS dataset to inform the dust masses per airport
- Pooling of Dirichlet parameters geographically
- Which airports should we empirically measure to best reduce the uncertainty in our posterior estimates?

"That's all Folks!"

Time-dependence extension for inverse problems

Modified setup:

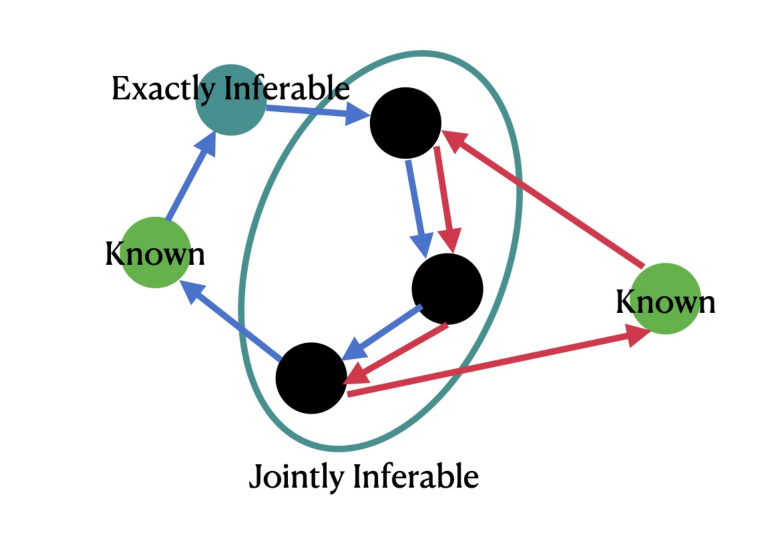
- i: specific airport
- T: total observations of flights over time
- Weighting later flights more than earlier

$$\tilde{x}_i = \sum_{t=1}^{T} \beta^{T-t} \mathbf{x}_t^i$$

Therefore

$$y = \frac{1}{n_{flights}} \sum_{flights;i} \tilde{x}_i$$

Which airports can be inferred exactly?



Toy forward model

- Assume each airport has a time-independent concentration vector, $\boldsymbol{x}_i$
- Final concentration vector $\boldsymbol{y}$

We look to consider the values of the $\boldsymbol{x}_i$'s for each engine. We first break down the $\boldsymbol{y}$ vector into a sum over the flights:

$$\boldsymbol{y} = \sum_i \frac{1}{(m_{i,in} + m_{i,out})} \sum_{flights;i} m_{i,in}\boldsymbol{x}_{i,in} + m_{i,out}\boldsymbol{x}_{i,out}$$

We further this by generating a toy problem by assuming that the contribution of the arrival vector is low and so $\boldsymbol{y}$ reduces to:

$$\boldsymbol{y} = \frac{1}{n_{flights}} \sum_{flights;i} \boldsymbol{x}_i$$

Toy forward model 2

From here we rewrite as a linear system $Ax = y$, we collate the x_i's into one by considering $\mathbf{x} = [\mathbf{x}_1; \mathbf{x}_2; ...; \mathbf{x}_{n_{airports}}]$ and we can break down A into a flat matrix determined by the Kronecker product: $A = \frac{1}{n_{flights}}[n_1, n_2, ...n_{n_{airports}}] \otimes \mathbb{I}_5$, where n_i is the number of visits to airport i in each engine's life.

We can consider the set $\mathcal{I} = \{i : n_i > 0\}$ and reduce A to $A = \frac{1}{n_{flights}}[n_j, j \in \mathcal{I}] \otimes \mathbb{I}_5$ and $\mathbf{x}$ to $\mathbf{x} = [\mathbf{x}_j; j \in \mathcal{I}]$.

This system is still under-determined (and sparse) but the pseudo-inverse *should* work to get a solution.