Boring but important disclaimers:

- If you are not getting this from the GitHub repository or the associated Canvas page (e.g. CourseHero, Chegg etc.), you are probably getting the substandard version of these slides Don't pay money for those, because you can get the most updated version for free at

  `https://github.com/julianmak/academic-notes`

  The repository principally contains the compiled products rather than the source for size reasons.
- Associated Python code (as Jupyter notebooks mostly) will be held on the same repository. The source data however might be big, so I am going to be naughty and possibly just refer you to where you might get the data if that is the case (e.g. JRA-55 data). I know I should make properly reproducible binders etc., but I didn't...
- I do not claim the compiled products and/or code are completely mistake free (e.g. I know I don't write Pythonic code). Use the material however you like, but use it at your own risk.
- As said on the repository, I have tried to honestly use content that is self made, open source or explicitly open for fair use, and citations should be there. If however you are the copyright holder and you want the material taken down, please flag up the issue accordingly and I will happily try and swap out the relevant material.

OCES 3301 :

basic Data Analysis in ocean sciences

Bonus Session: Machine Learning

Outline

(Just overview here; for actual content see Jupyter notebooks)

- a very loose introduction to Machine Learning (ML)

  $\rightarrow$ example with argo data

  $\rightarrow$ unsupervised ML example: clustering analysis

  $\rightarrow$ supervised ML example: neural networks

Some propaganda to start with

ML algorithms are:

- algorithms + tools, and that's it

  $\rightarrow$ very powerful, but context dependent
- usually black box

  $\rightarrow$ it can work wonderfully / fail spectacularly, but you don't necessarily know why...

!!! prudent to do sanity checks!

Figure: Hermeowus Mora, disciple of Hermaeus Mora the Daedric prince of knowledge and memory

Cursed example: image recognition/generation

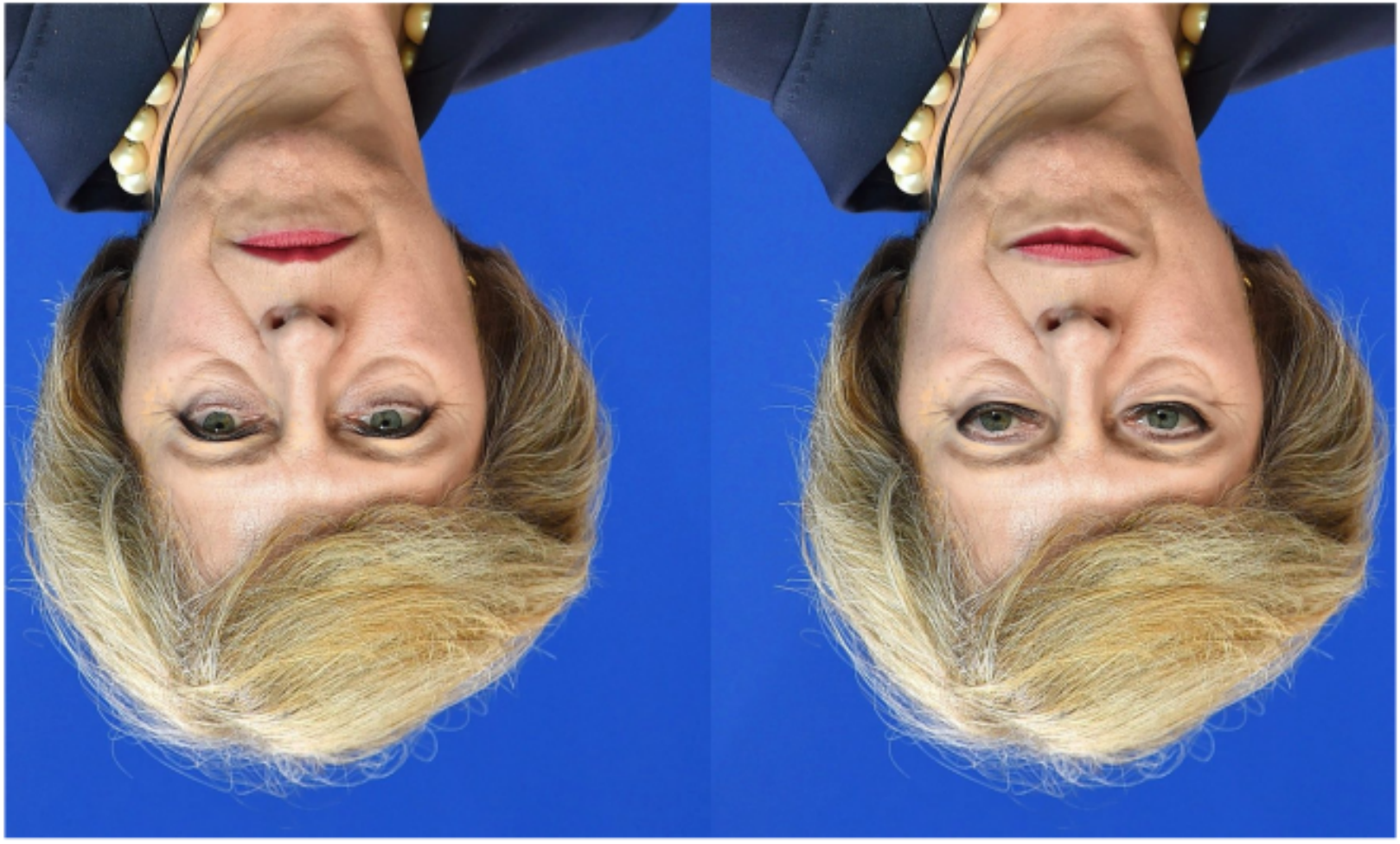


Figure: Cursed beast?

Cursed example: image recognition/generation

Figure: Cursed

Machine learning + regression/optimisation (see 03 + 04)

For X the input, y the output, f the model, where

$$y = f(X),$$

the aim is to seek f such that we minimise something like

$$J = \sum_i (y_i - f(X_i))^2$$

- ML in a nutshell follows the same principle

  $\rightarrow$ algorithms are different (e.g. nonlinear, network based, different optimiser)

Training, Validation, Testing data

Normally split (X, y) into

- training data $(X_{\text{train}}, y_{\text{train}})$ (most data should be here)

  $\rightarrow$ exposed to ML algorithms for training the model

  $\rightarrow$ used to compute misfits or loss function
- validation data $(X_{\text{val}}, y_{\text{val}})$

  $\rightarrow$ exposed to ML algorithms to tune model hyperparameters and/or model selection
- test data $(X_{\text{test}}, y_{\text{test}})$

  $\rightarrow$ **NOT** exposed to ML algorithm

  $\rightarrow$ used to test performance of model

!!! sometimes "validation" and "test" are swapped

Unsupervised vs. supervised

- unsupervised ML is where data is unlabelled, and algorithm picks out features by themselves

  $\rightarrow$ e.g. PCA (so EOFs), clustering, some examples of neural networks

Figure: Cursed cats/dogs (?) from PCA. Figure adapted from Fig. 10 of Brunton, Brunton, Proctor & Kutz (2013).

Unsupervised vs. supervised

- supervised ML is where data is labelled, and algorithm fits model between input and output

  $\rightarrow$ often want this for prediction purposes

  $\rightarrow$ e.g. (multi-)linear regression, some examples of neural networks

- other characterisations (e.g. semi-supervised, reinforcement)

Unsupervised vs. supervised

Figure: Various entries from the “animal or things” meme, as found on the internet.

Argo (see also lec 19 of OCES 2003)

- A CTD gets
  - → conductivity to get S
  - → temperature for T
  - → it really measures p to get depth
  - → can put other sensors on (e.g. pH, oxygen, etc.)
- argo system consists of CTDs that floats around the ocean

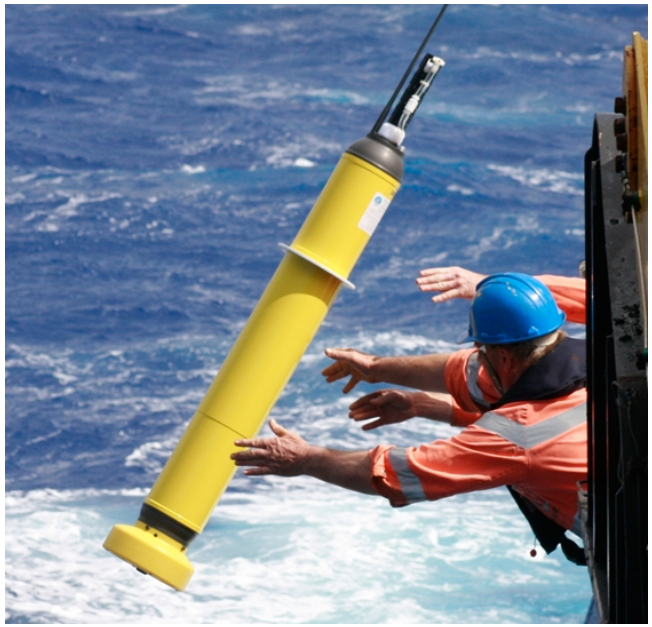


Figure: An Argo float being thrown off a ship. Image from NOAA.

Argo (see also lec 19 of OCES 2003)

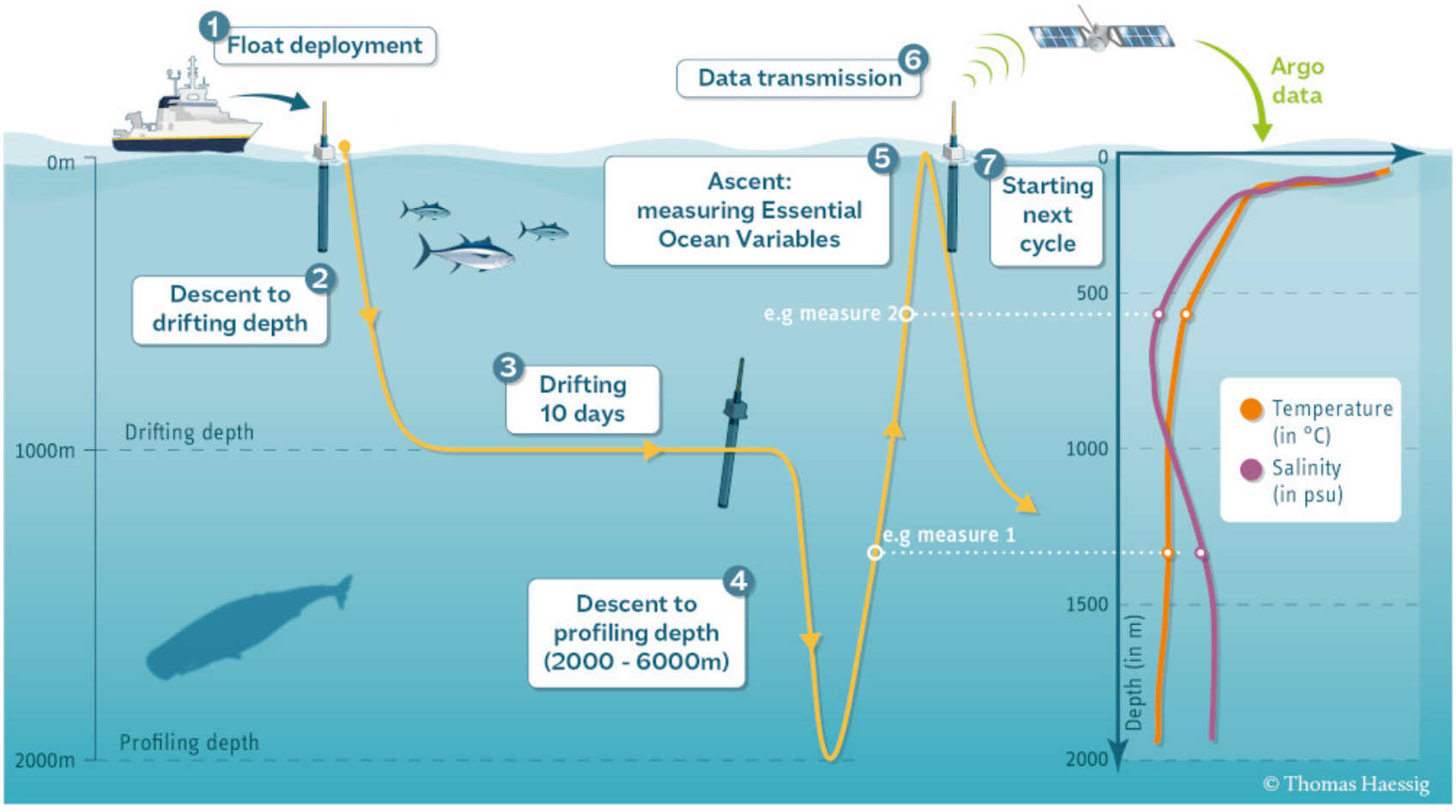


Figure: Argo float cycle schematic. From `argo.ucsd.edu`

Argo (see also lec 19 of OCES 2003)

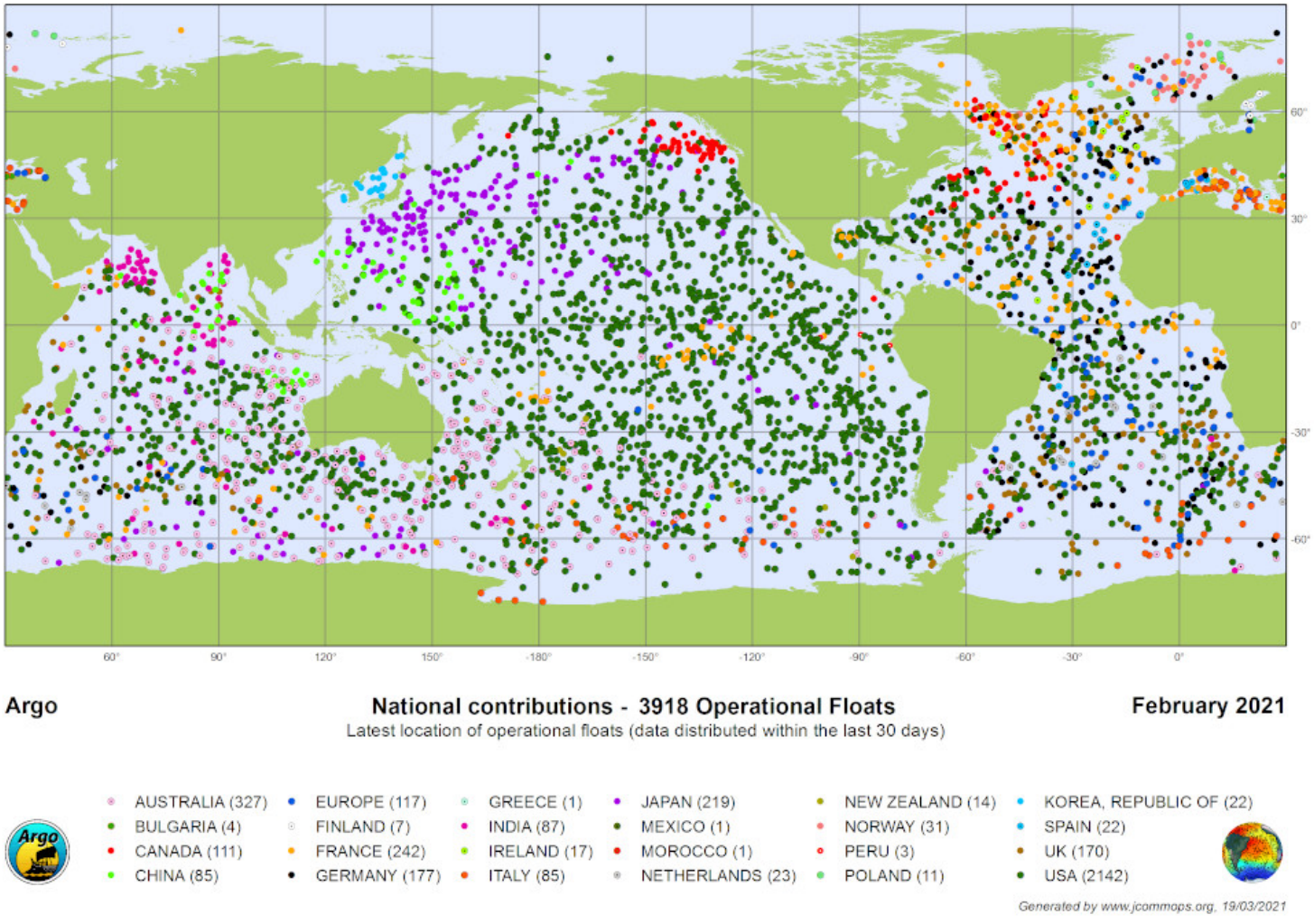


Figure: Argo locations as of Feb 2021. Note the dots are enhanced in size, so coverage is not as dense as it seems. From `argo.ucsd.edu`

Argo (see also lec 19 of OCES 2003)

▸ Dimensions: (**DEPTH**: 302, N_PROF: 128910)

▾ Coordinates:

DEPTH	(DEPTH)	float32	0.0 -5.0 -10.0 ... -1500.0 -1505.0
LATITUDE	(N_PROF)	float32	dask.array<chunksize=(67010,), meta=n...
LONGITUDE	(N_PROF)	float32	dask.array<chunksize=(67010,), meta=n...
TIME	(N_PROF)	datetime64[ns]	dask.array<chunksize=(64455,), meta=n...

▾ Data variables:

BRV2	(N_PROF, DEPTH)	float32	dask.array<chunksize=(67010, 302), met...
DBINDEX	(N_PROF)	float64	dask.array<chunksize=(67010,), meta=n...
PSAL	(N_PROF, DEPTH)	float32	dask.array<chunksize=(67010, 302), met...
SIG0	(N_PROF, DEPTH)	float32	dask.array<chunksize=(67010, 302), met...
TEMP	(N_PROF, DEPTH)	float32	dask.array<chunksize=(67010, 302), met...

▸ Attributes: (12)

Figure: Argo dataset in zarr format opened as a xarray object.

- argo data here given in zarr format
  → need `zarr` package, can open data through xarray
  → ungridded data here
  → see also `argopy` package

Argo (see also lec 19 of OCES 2003)

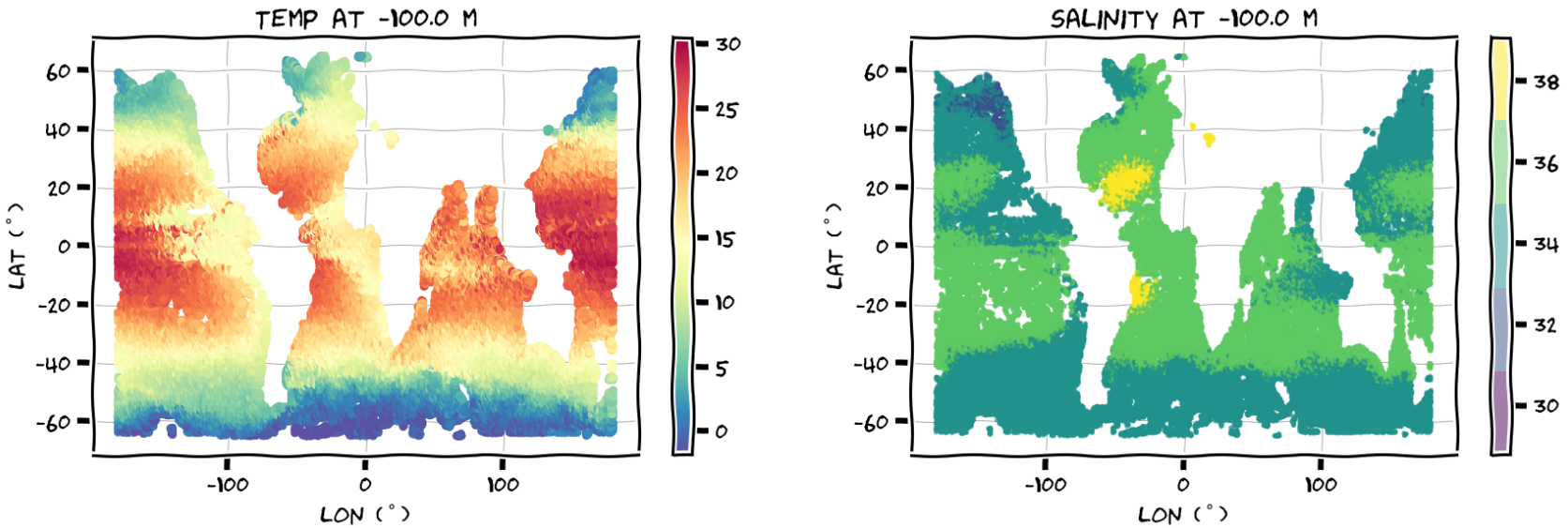


Figure: Argo (in-situ) temperature and (practical) salinity at some fixed depth as a scatter plot coloured by data entry.

- ungridded data, each point is an entry
  $\rightarrow$ scatter plot, with dot coloured by data

Argo: clustering

- we know different watermasses have different properties
  → e.g. NADW is more salty
  → unsupervised learning?

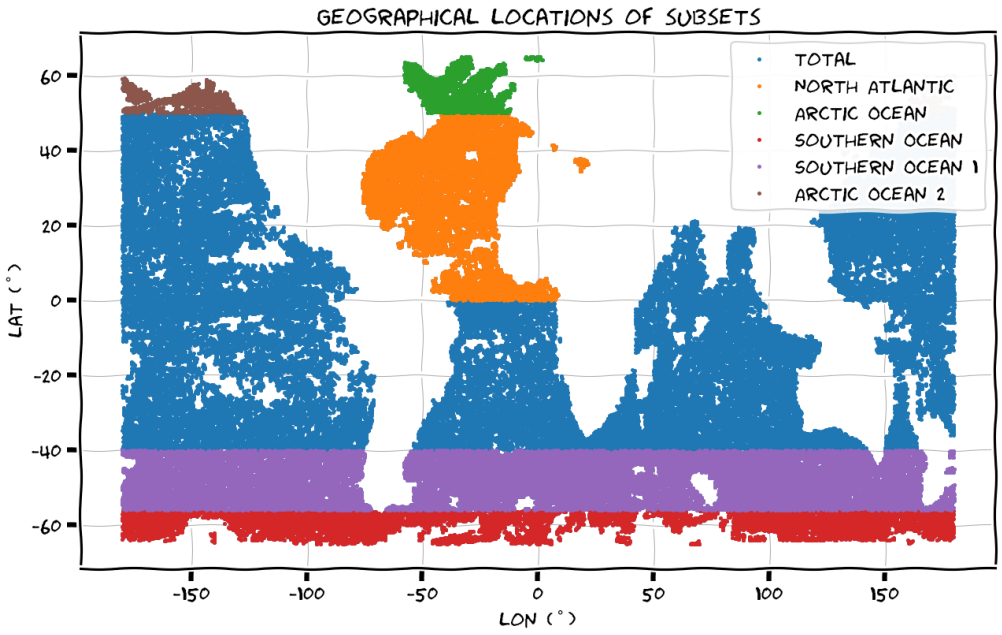


Figure: Artificial clustering.

Argo: clustering

- we know different watermasses have different properties
  → e.g. NADW is more salty
  → unsupervised learning?

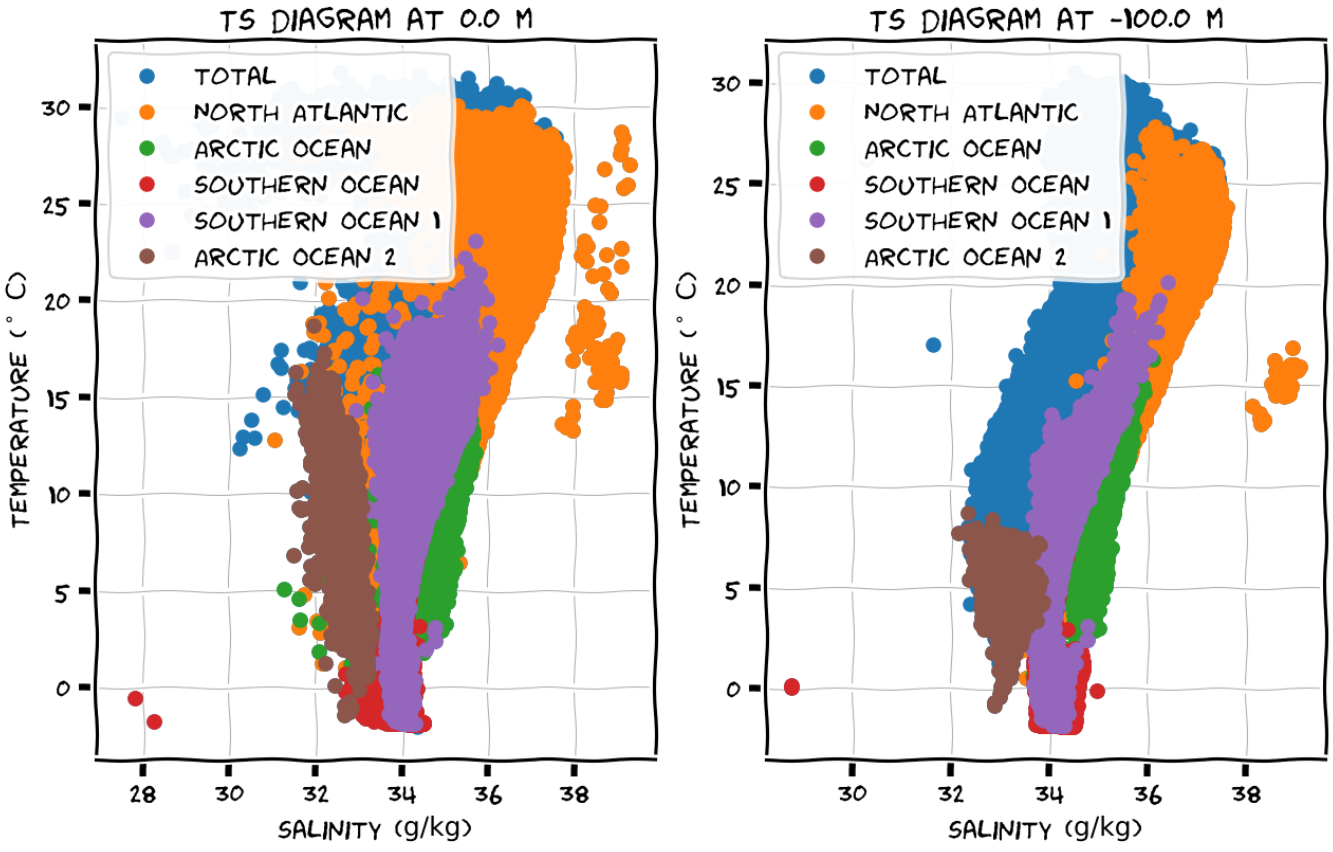


Figure: Artificial clustering as above, but in *TS*-diagram.

Argo: clustering

- one example is *k*-means

  → partition data and find means

  → move partitions slightly and compute new means

  → iterate on partitions such that distance to partition means are minimised (finds local minimums)

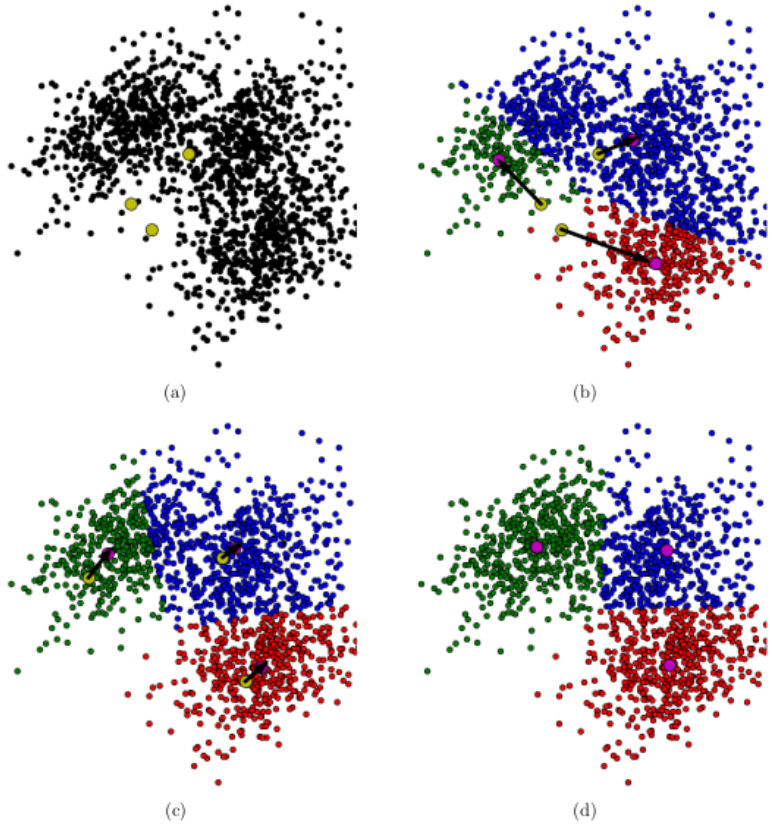


Figure: Demonstration of *k*-means algorithm. Diagram taken from Machine Learning course of Christoph Haase and Varun Kanade at University of Oxford.

Argo: clustering

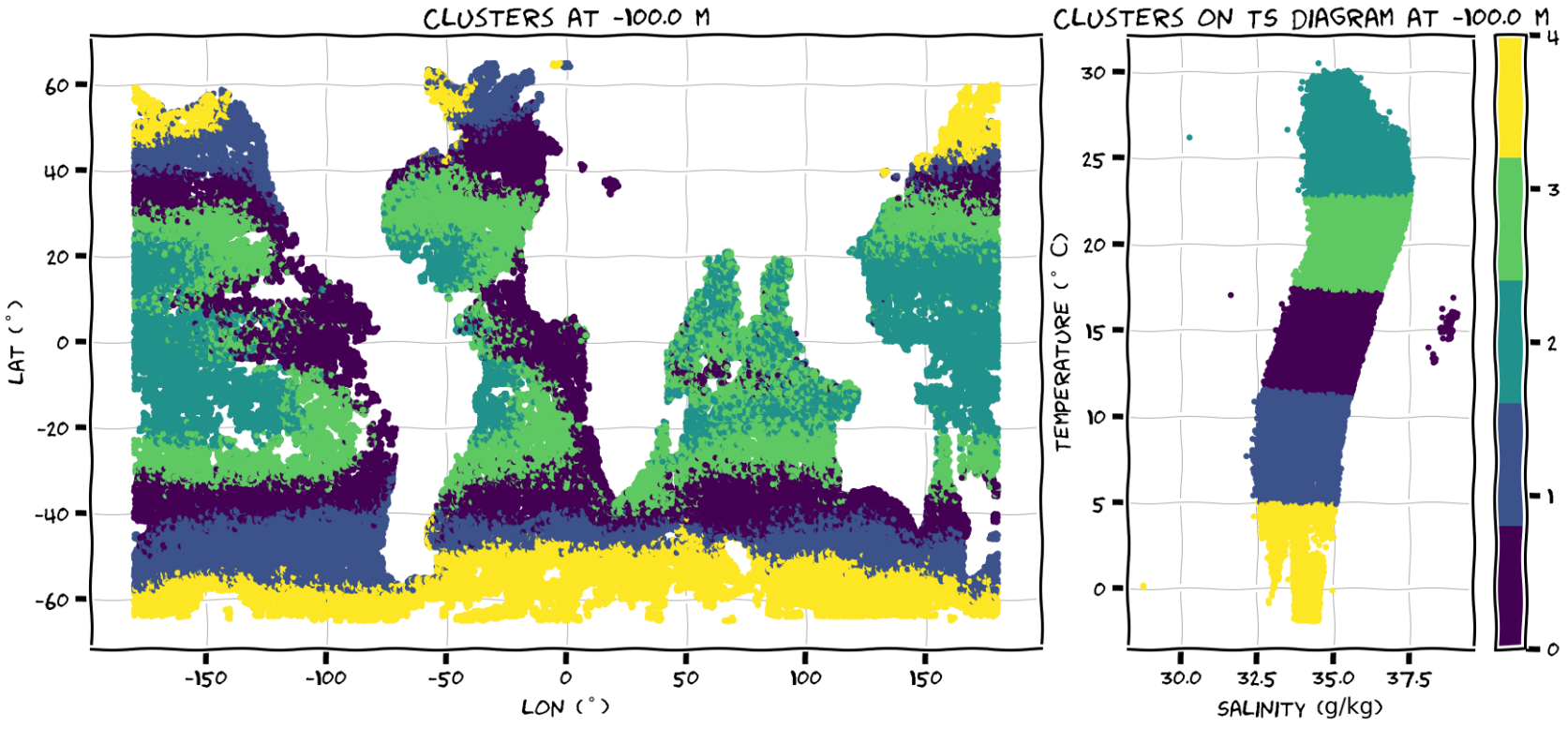


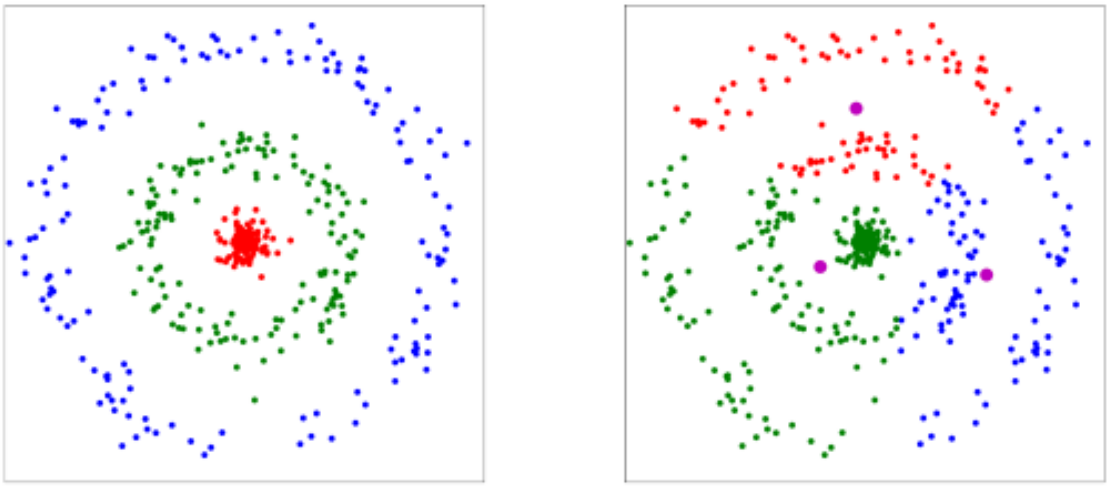
Figure: Clustering from *k*-means algorithm.

- *k*-means in *TS*-space really
- some physical rationalisation possible

!!! didn't standardise data here (probably should have)

Argo: clustering

- example where *k*-means can fail



(a) Data with three clusters as concentric circles

(b) Output of *k*-means algorithm

Figure: Demonstration of failure of *k*-means algorithm. Diagram taken from Machine Learning course of Christoph Haase and Varun Kanade at University of Oxford.

Argo: clustering

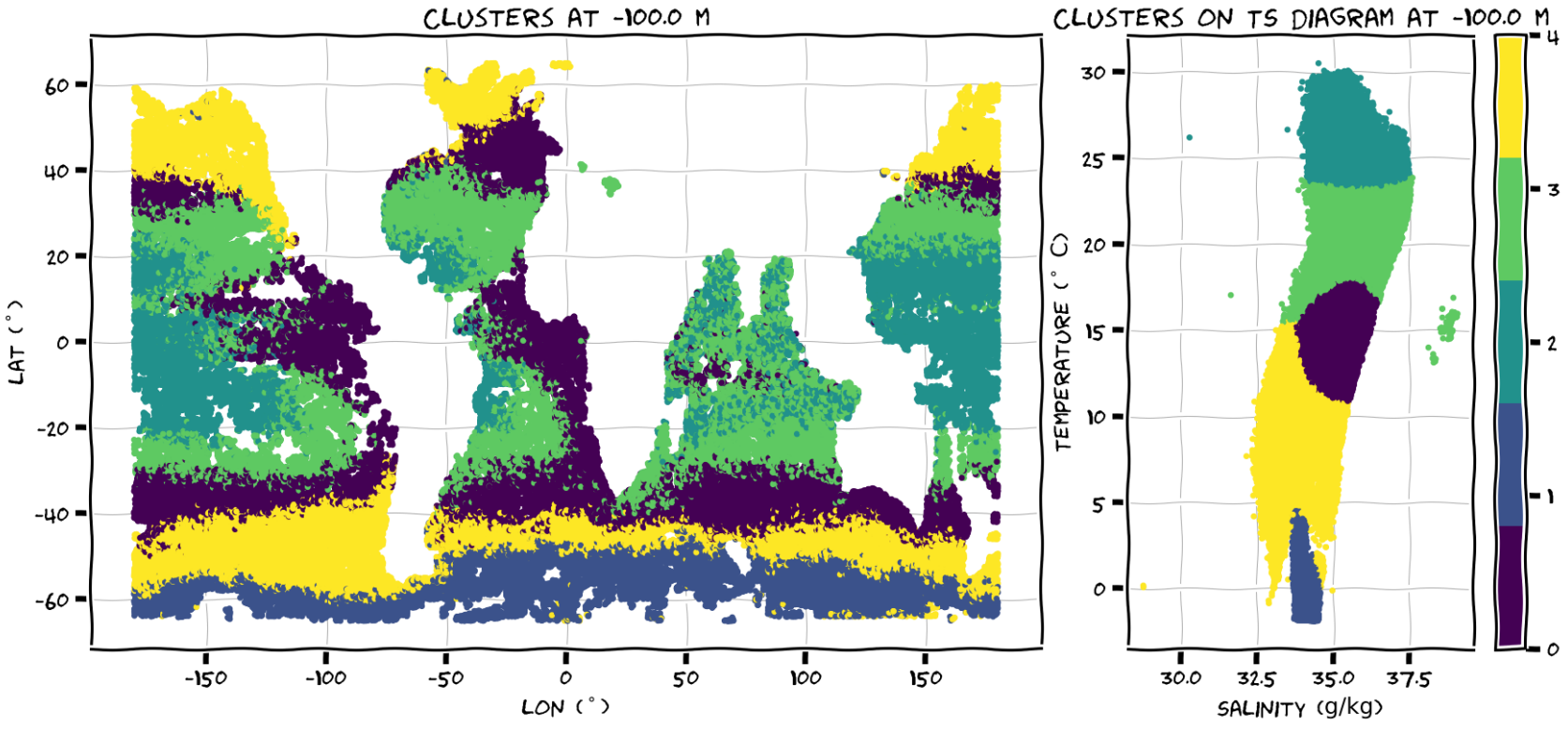


Figure: Clustering from Gaussian Mixture Model.

- similar to above but with some subtle differences
- GMM used in oceanography before (e.g. Jones *et al.*, 2019, in Southern Ocean)

!!! didn't standardise data here (probably should have)

Argo: neural network

- suppose we want to predict salinity from temperature
  → more for demonstration really...
  → split data first (`sklearn.train_test_split`)

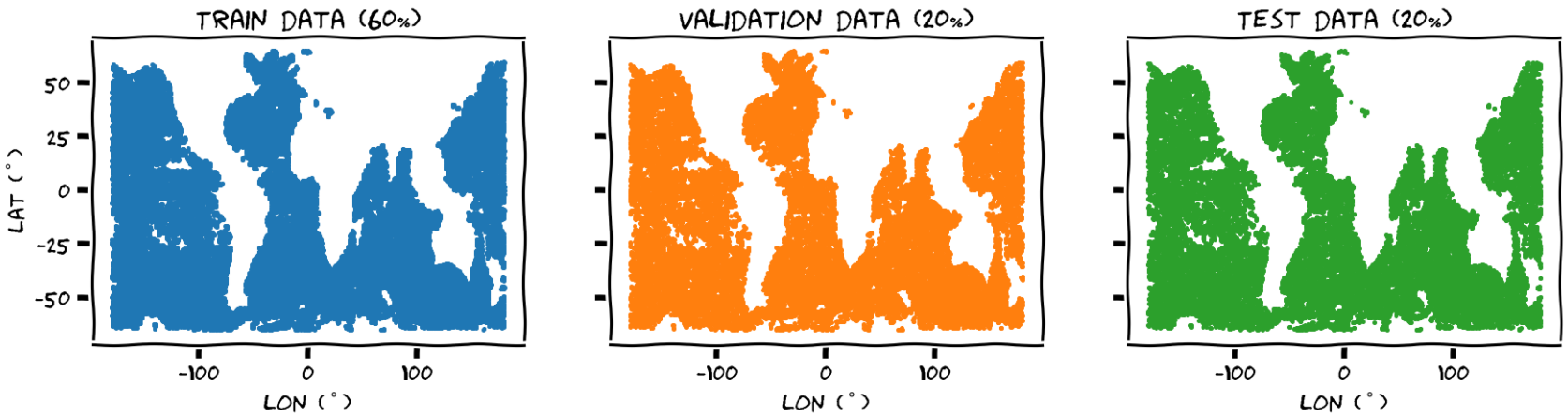


Figure: Splitting of argo data into training:validation:test as 60:20:20.

Argo: neural network

- linear regression?

  $\rightarrow$ standardise data

  $\rightarrow$ train with training data (could in principle use everything)

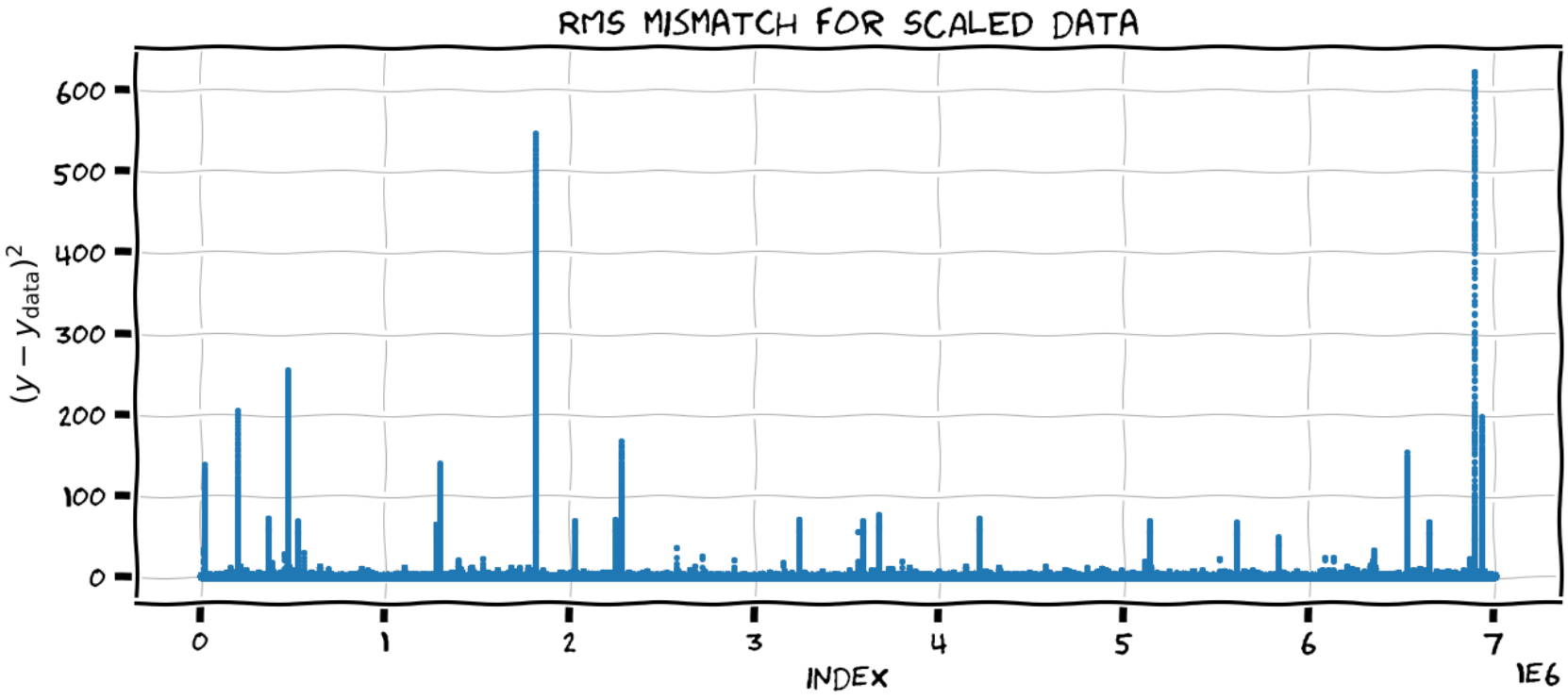
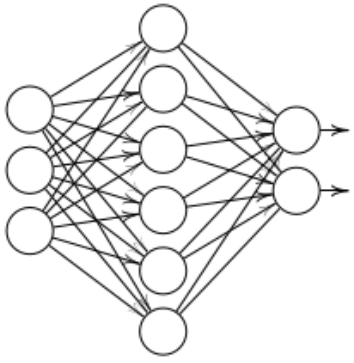
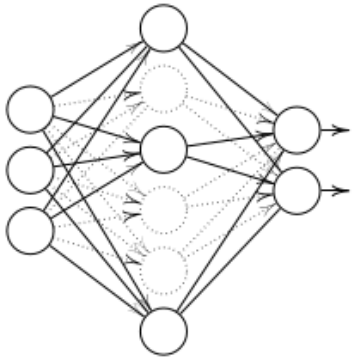


Figure: Root-mean-square loss against index for scaled data, so anything larger than 1 is pretty bad.

Argo: neural network



(a) Neural Network



(b) Neural Network with Dropout

Figure: Schematic of neural network. Diagram adapted from Machine Learning course of Christoph Haase and Varun Kanade at University of Oxford.

- a neural network has

  → nodes containing features or transformation rules

  → links linking the nodes

  → weights tagged with links specifying weighting or transition probability going from one node to another
- simple case would be adjusting weights to minimise the mismatch / loss function

  → could in principle adjust features in nodes etc.

  → drop off procedure as a stabiliser

Argo: neural network

- train model with training and validation data
  $\rightarrow$ data has been standardised here
- model trained over 30 epochs
  $\rightarrow$ think 30 complete passes/iterations

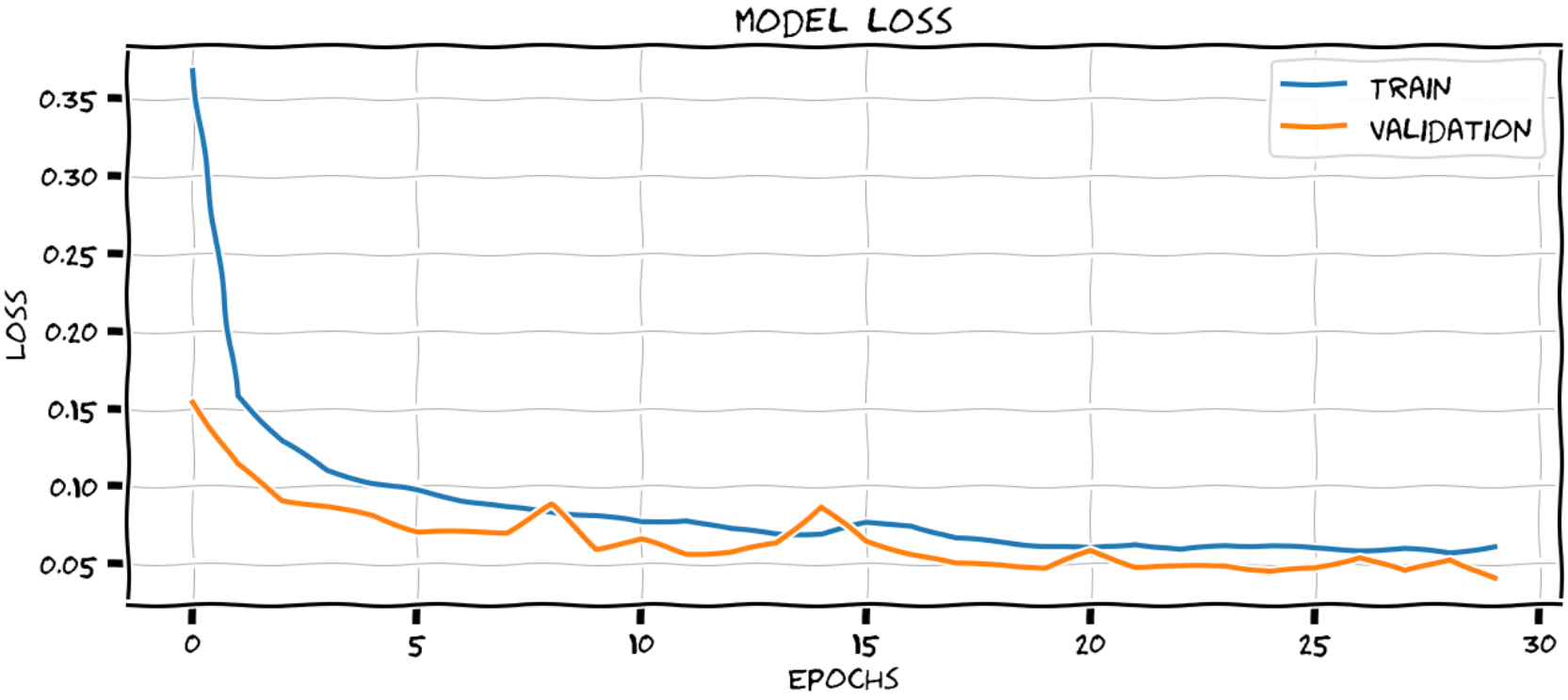


Figure: RMS loss against epoch. Note the RMS loss is not zero (and we don't expect it to be).

Argo: neural network

- model takes an input depth varying in-situ temperature and returns a salinity profile

  $\rightarrow$ three random realisations below, with scaling inverted

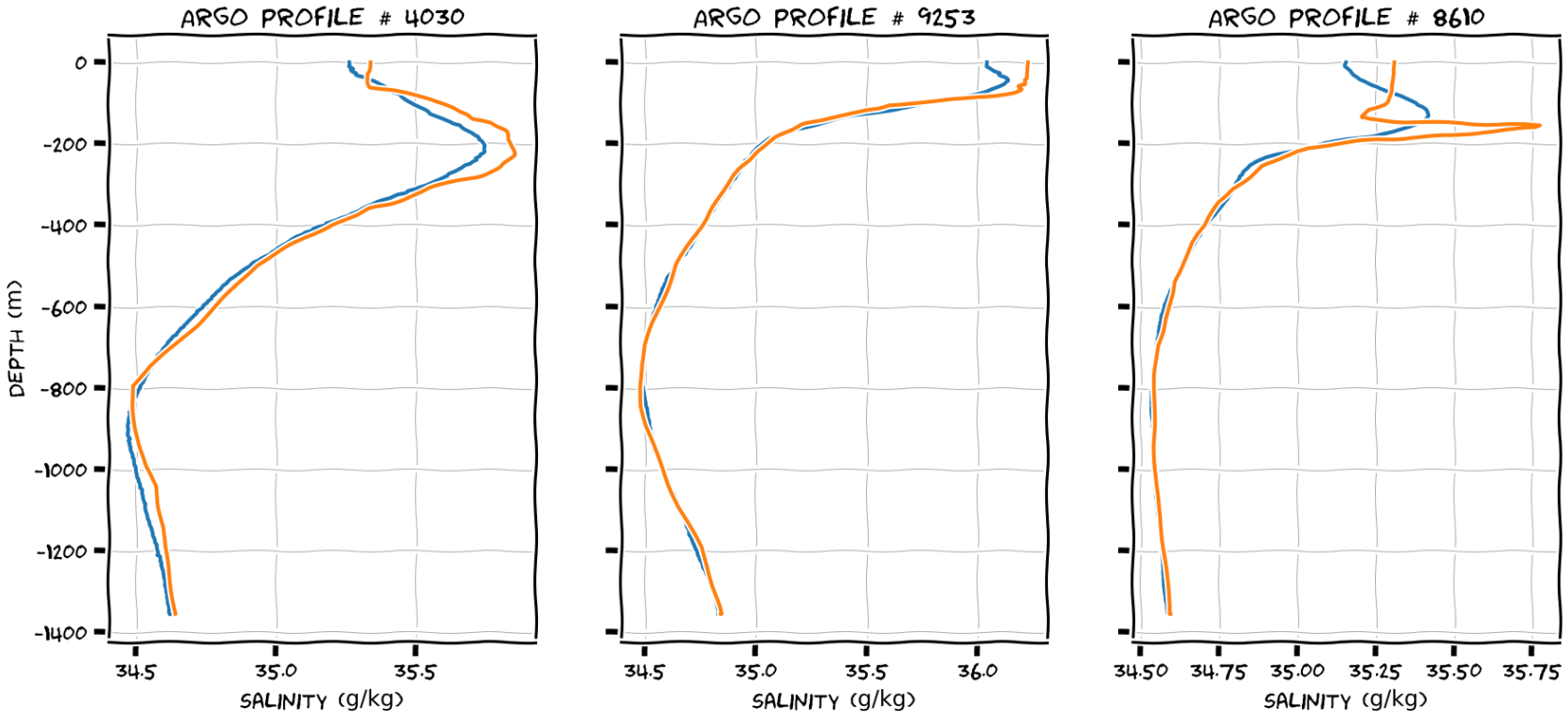


Figure: Three examples of using the trained neural network.

Argo: neural network

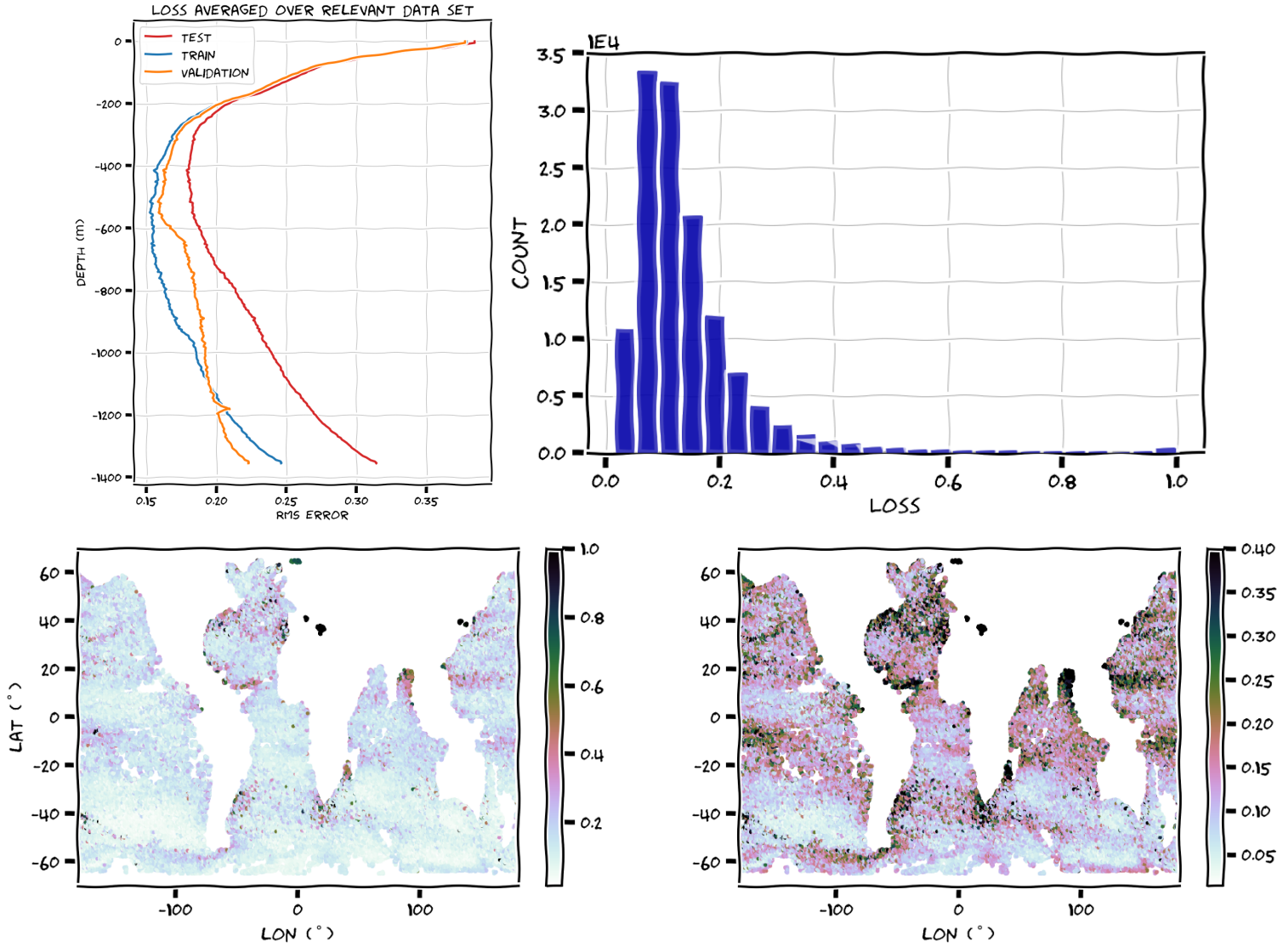


Figure: Some summary plots of the loss.

Argo: neural network

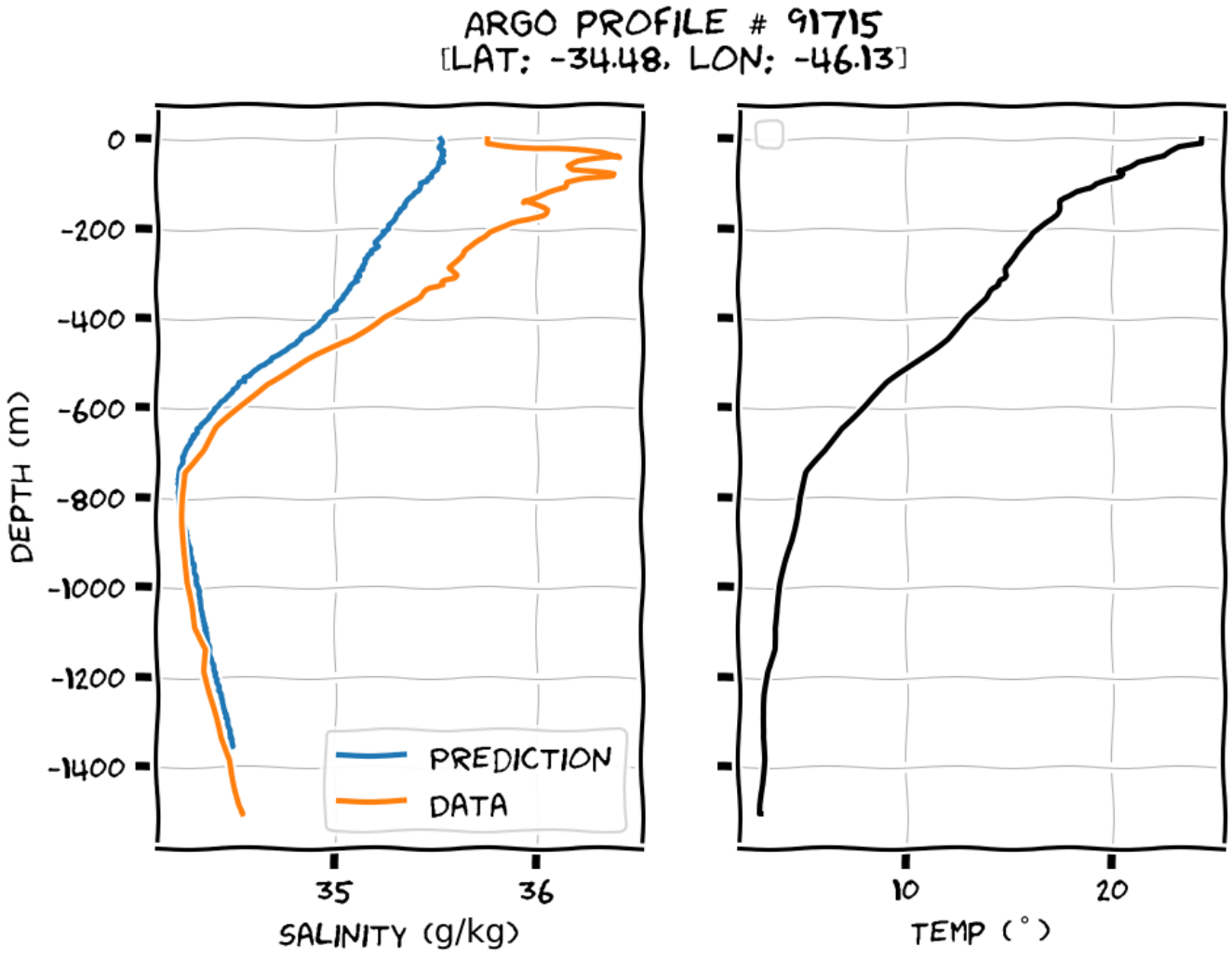


Figure: Example of a case with particularly high error.

Jupyter notebook

go to bonus Jupyter notebook (with thanks to Fei Er) to get some code practise

- different ways of reading the argo data
- different algorithms to try
- different questions to ask
- different datasets (e.g. Iris and/or Penguin)?
- . . .

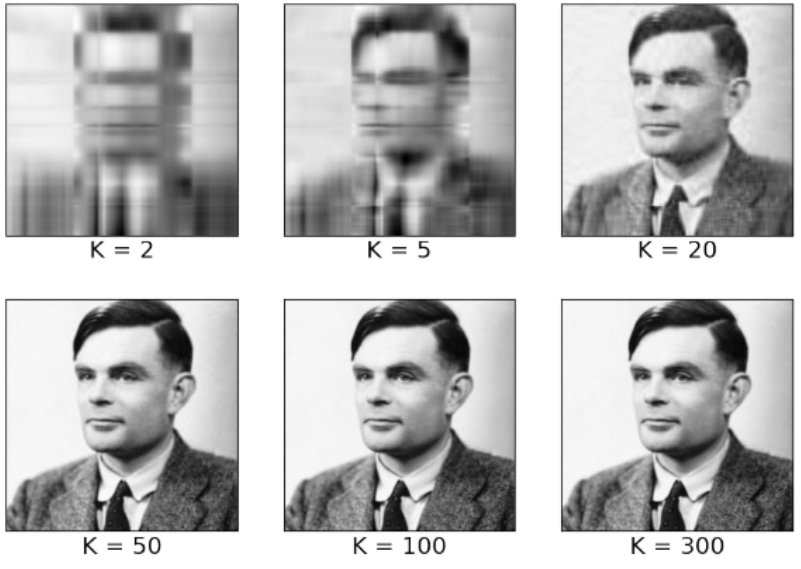


Figure: Image reconstruction (who is this person?) Neural network with data from PCA? (cf. exercise in 04, linear regression of data from PCA)