Short-term forecast of the geomagnetic secular variation using recurrent neural networks trained by Kalman filter

Sho Sato , Hiroaki Toh

Division of Earth and Planetary Sciences, Graduate School of Science,
Kyoto University, JAPAN

Wed, 18 Sep. 09:20-09:40 | IUGONET 研究集会 @ ハイブリッド

This work is supported by **ROIS-DS-JOINT Grant Number 008RP2024**.

OVERVIEW

1. INTRODUCTION

- ⦿ Background : *Geomagnetic Secular Variation* and *Geomagnetic Jerk*
- ⦿ Previous Research : Data Assimilation = Geodynamo SIM x Observed Data

2. METHODOLOGY

- ⦿ Basics of Machine Learning and *Recurrent Neural Networks*
- ⦿ Overfitting problem in the Backpropagation algorithm
- ⦿ The extended Kalman filtering approach to train *RNNs*

3. RESULTS SO FAR

- ⦿ Hindcast (training: 2004.75 – 2014.50 → validation: 2014.75 – 2019.50)
- ⦿ Forecast (training: 2014.37 – 2024.13 → validation: 2024.37 – 2029.13)

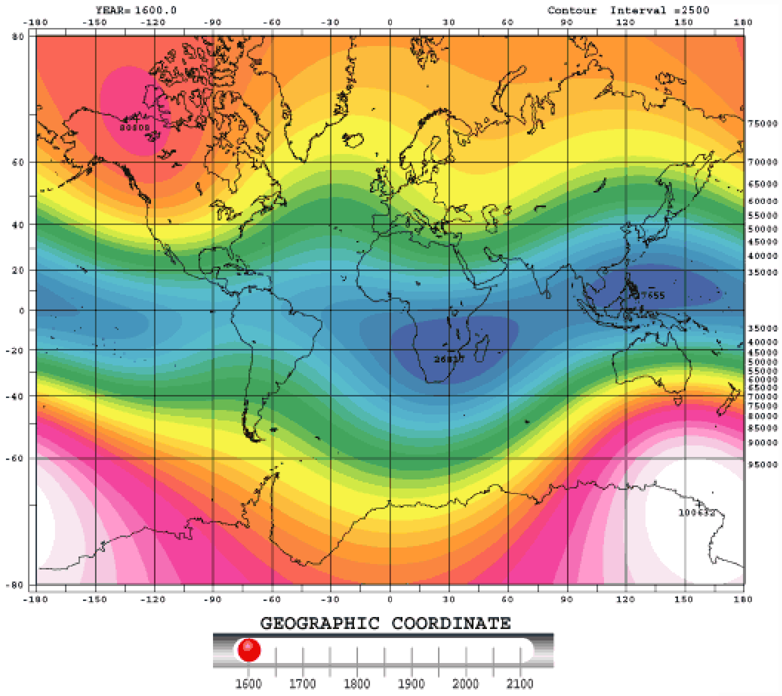
4. RESEARCH PLAN

- ⦿ Conclusion & Road Map

INTRODUCTION | Background

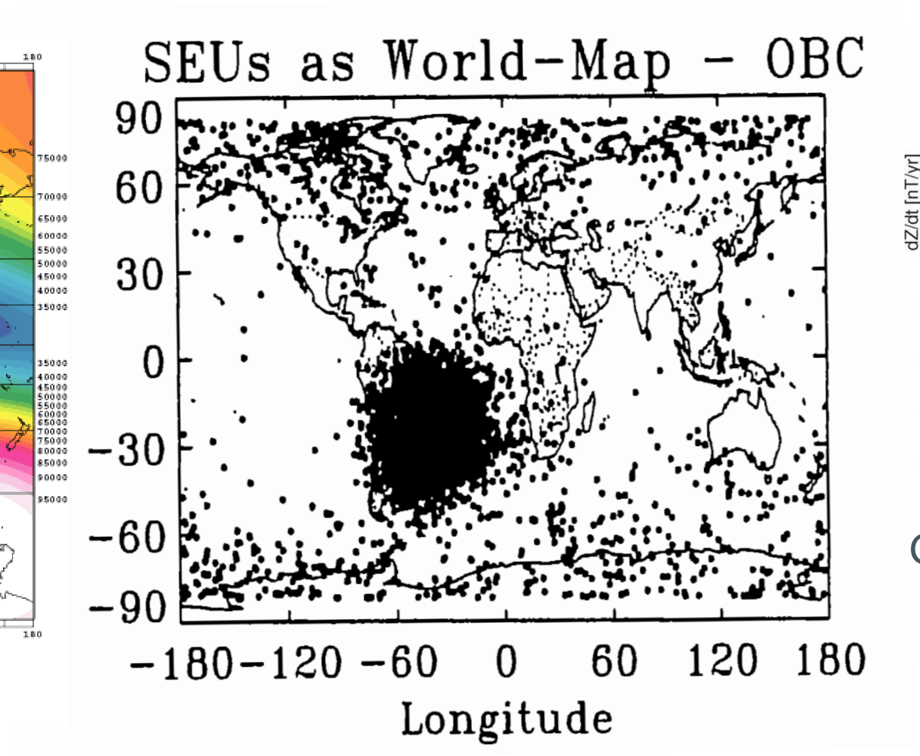
Geomagnetic ***Secular Variation***

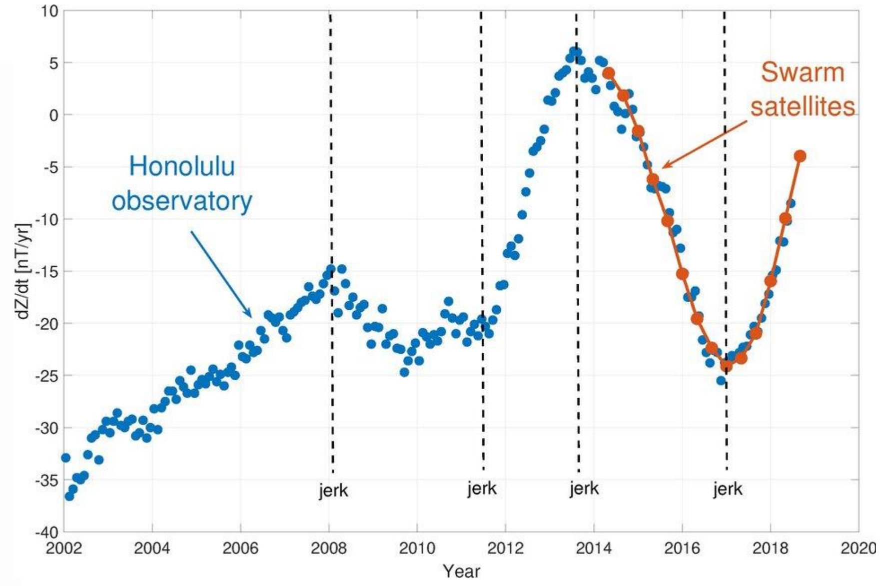
(Image: WDC for Geomagnetism, Kyoto*1)



Location of UOSat-2 when ***Single-event Upsets*** were detected in its OBC memory during 1989

(Image: Daly et al, 1996*2)





Geomagnetic ***Jerks*** (= Sudden Acceleration)

(Image: ESA/DTU*3)

The ***Gauss coefficients*** and their **derivatives** have been reported every 5 years as ***International Geomagnetic Reference Field.***

**Gauss coefficients* (g_0^1 , g_1^1 , h_1^1 , ...) = Spherical harmonic coefficients of the Geomagnetic scalar potential, equivalent to multipole moments.

1. https://wdc.kugi.kyoto-u.ac.jp, 2. Daly et al (1996), "Space Environment Analysis: Experience and Trends" vol.392. p.15., 3. Swarm helps explain Earth's magnetic jerks.

INTRODUCTION | Previous Research

IGRF-13 candidate SV models= **Data Assimilation** (**Geodynamo SIM** x Observation)

		Forecast Error (RMSE [nT])	Hindcast	Simulation Model	Observational Data
Sanchez et al. (2020)[*1]		$50 \sim 100$	$1840 \sim 2020$	**3D dynamo** x 100 ens.	COV-OBS.x1 Kalmag model
Minami et al. (2020)[*2]		$100.9 \sim 228.5$	$2004 \sim 2019$	**MHD dynamo** x 1000 ens.	MCM model
Tangborn et al. (2021)[*3]		$O(10^{1.5} \sim 10^2)$	$1590 \sim 2015$	**NASA GEMS** x 400 ens.	gufm1, CM5, CM6

MOTIVATION : Can ML models be a new approach to forecast SV without simulations?

*1. doi: 10.1186/s40623-020-01279-y , *2. doi: 10.1186/s40623-020-01253-8 , *3. doi: 10.1186/s40623-020-01324-w.

INTRODUCTION | ML / DL in Geophysics

Reviews of Geophysics

ADVANCING EARTH AND SPACE SCIENCE

REVIEW ARTICLE
10.1029/2021RG000742

Deep Learning for Geophysics: Current and Future Trends

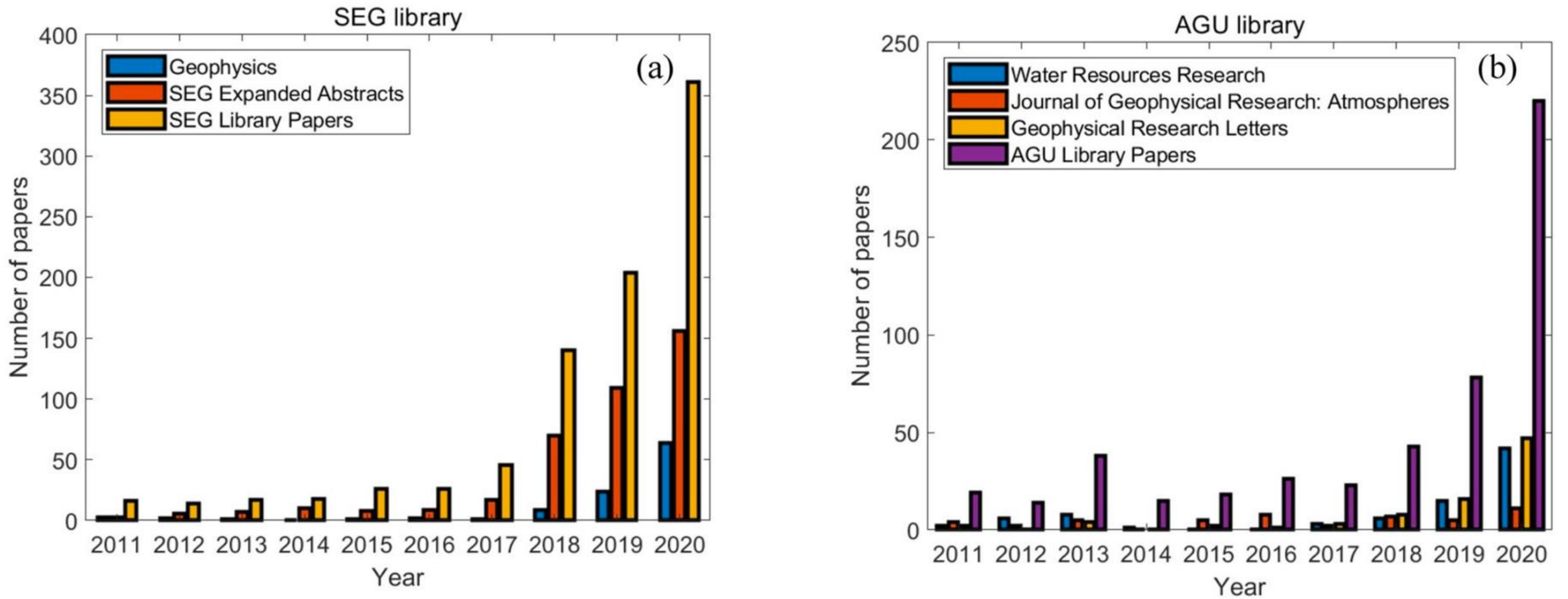


Figure 3. (a) and (b) are statics of artificial intelligence (AI)-related papers in SEG Library and AGU Library. In (a), Geophysics means the flagship journal of SEG. SEG Expanded Abstracts means the Expanded Abstracts from SEG annual meeting. SEG Library papers mean the papers founded in the SEG digital library. In (b), the first three captions in the legend are the names of top journals in AGU. The fourth caption in the legend represents the papers founded in the AGU digital library.

"The geophysical community has shown great interests in DL in recent years. Figure 3 shows the published papers related to artificial intelligence in two major geophysical unions, that is, society of exploration geophysics (SEG) and American geophysical union (AGU). A clear exponential growth is observed in both libraries due to the use of DL techniques. Moreover, DL has also provided several astonishing results to the geophysical community."

Yu & Ma, 2021. (https://doi.org/10.1029/2021RG000742)

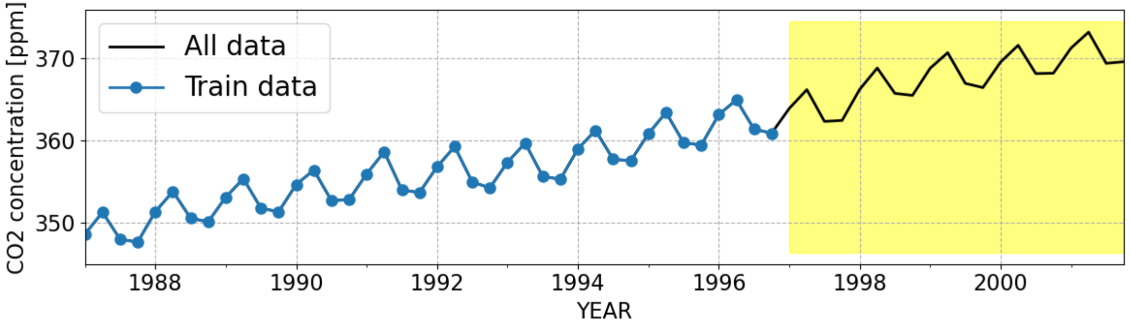
METHOD | What is ML training? = Parameter estimation

Example: **AR(4) model**

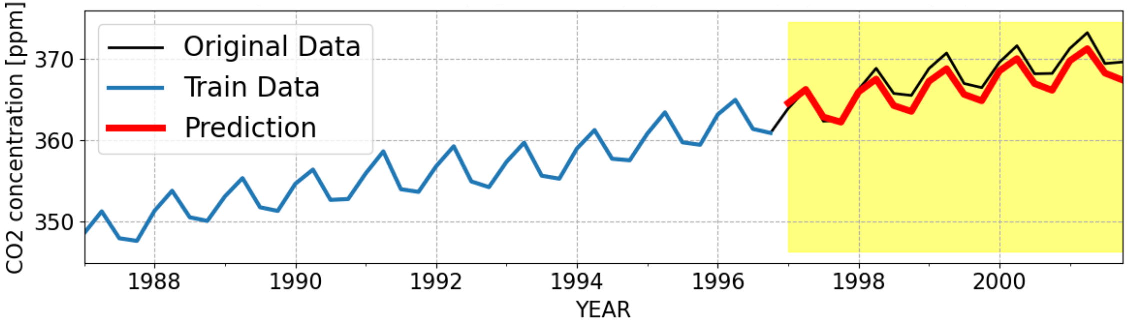
$$y_t = \boldsymbol{b} + \boldsymbol{w_1} y_{t-1} + \boldsymbol{w_2} y_{t-2} + \boldsymbol{w_3} y_{t-3} + \boldsymbol{w_4} y_{t-4}$$

(Trainable Parameter: $\boldsymbol{b}$ (bias) , $\boldsymbol{w_1} \sim \boldsymbol{w_4}$ (weight))

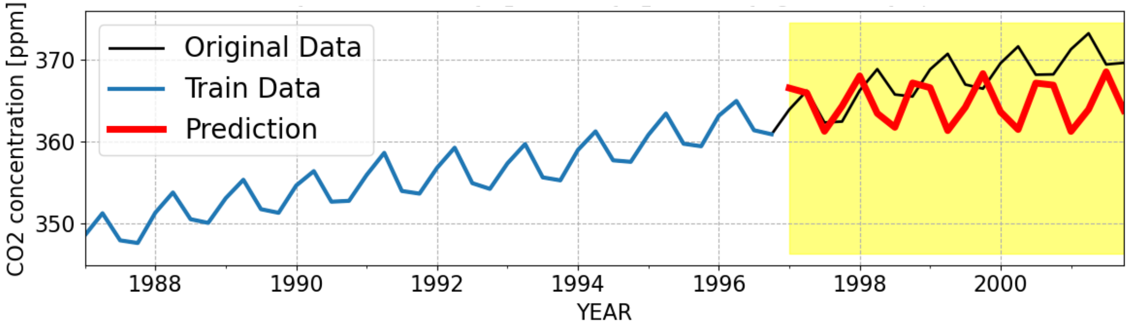
Original Data (CO_2 at Mauna Loa, Hawaii)



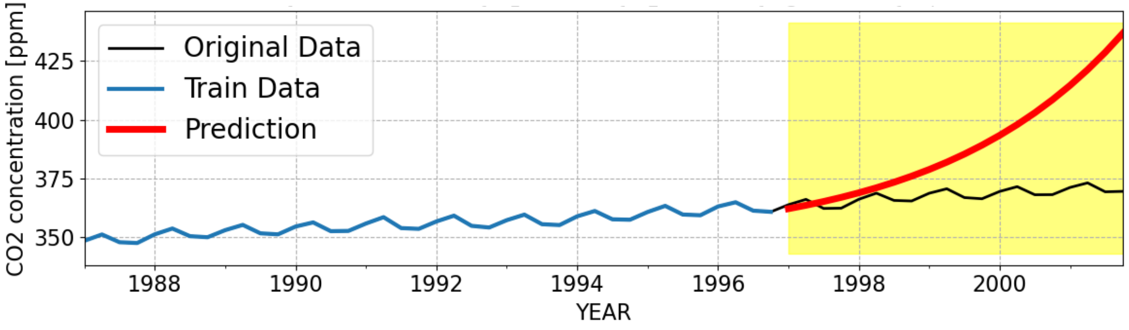
$$y_t = \mathbf{0.06} + \mathbf{0.13} y_{t-1} - \mathbf{0.13} y_{t-2} + \mathbf{0.12} y_{t-3} + \mathbf{0.86} y_{t-4}$$



$$y_t = 0.60 + 0.30 y_{t-1} - 0.80 y_{t-2} + 0.70 y_{t-3} - 0.10 y_{t-4}$$



$$y_t = 0.00 + 1.10 y_{t-1} + 0.00 y_{t-2} + 0.00 y_{t-3} + 0.00 y_{t-4}$$



METHOD | Why RNN?

Model		AR	VAR	BPTT-RNNs		Minami et al. (2020) (4D-EnVar)
Average $\sqrt{dP}$		34.48 nT	39.58 nT	**33.43 nT**		---
5-year $\sqrt{dP}$		88.05 nT	98.33 nT	**85.83** – 137.74 nT		100.9 – 228.5 nT
Remarks		---	---	$\sqrt{dP}$ depends on the initial $\mathbf{w}_0$ and the # of itr. of BPTT		MHD dynamo(x1000) Depends on the assim. window

Sato & Toh (2024.04; EGU @Austria): https://doi.org/10.5194/egusphere-egu24-16981.

METHOD | RNN architecture

Elman Network

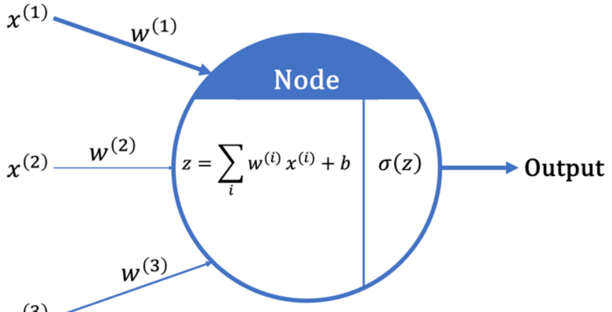
RNN cell

$$\mathbf{h}(t_i) = \tanh\left(\mathbf{W}^{(\mathrm{in})} \cdot \mathbf{x}(t_i) + \mathbf{W}^{(\mathrm{rec})} \cdot \mathbf{h}(t_{i-1}) + \mathbf{b}^{(\mathrm{rec})}\right)$$

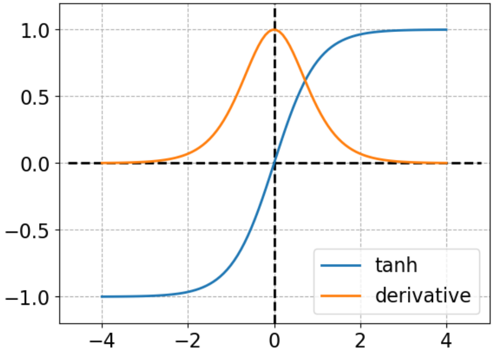
Affine (Dense)

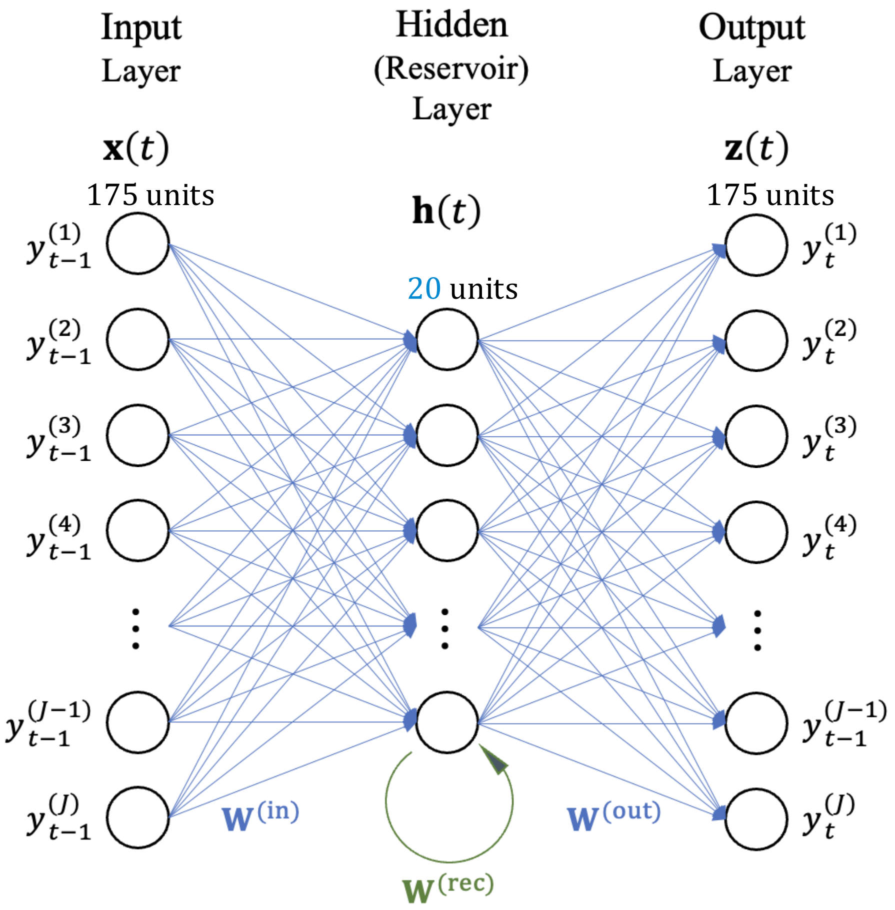
$$\mathbf{z}(t_i) = \mathbf{W}^{(\mathrm{out})} \cdot \mathbf{h}(t_i) + \mathbf{b}^{(\mathrm{out})}$$

$$(\text{input: } \mathbf{x}(t_i) = [\,\ddot{g}_n^{\,m}(t_{i-1})] \longrightarrow \text{output: } \mathbf{z}(t_i) = [\,\ddot{g}_n^{\,m}(t_i)]).$$



Activation: tanh





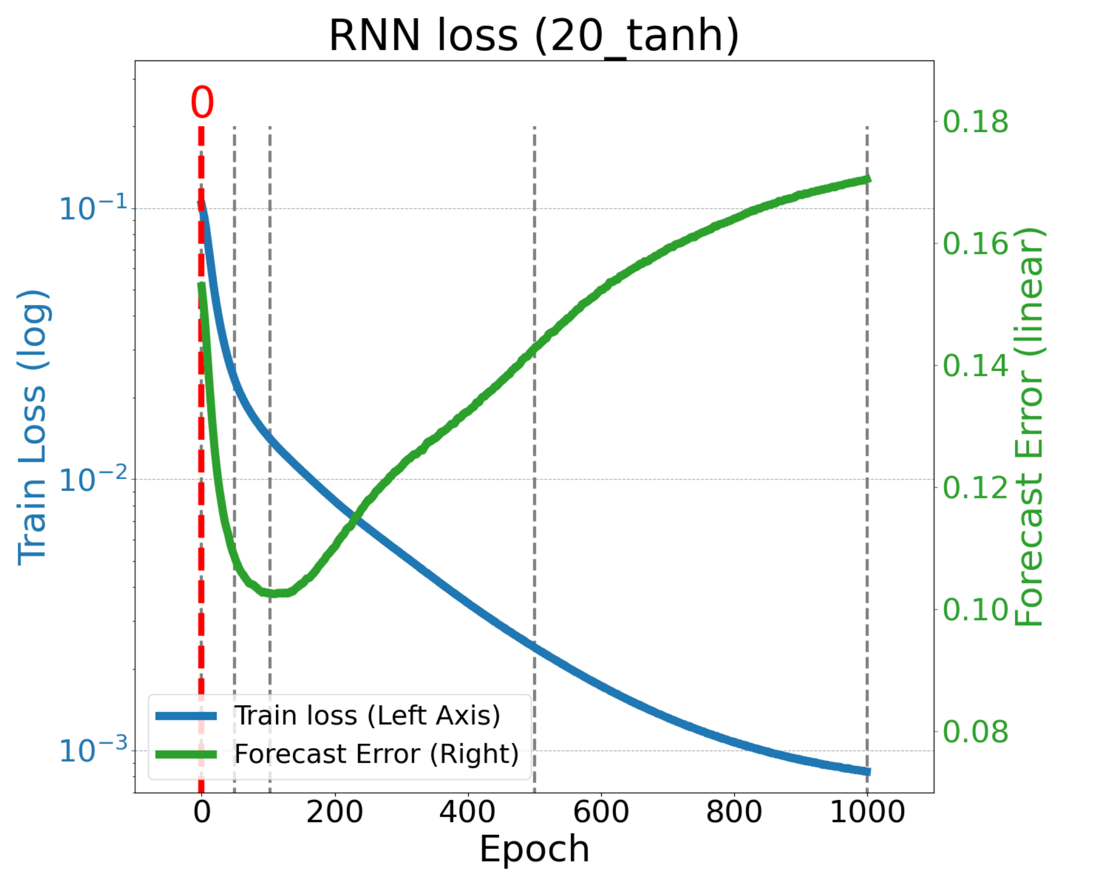
Elman (1990): https://doi.org/10.1207/s15516709cog1402_1.

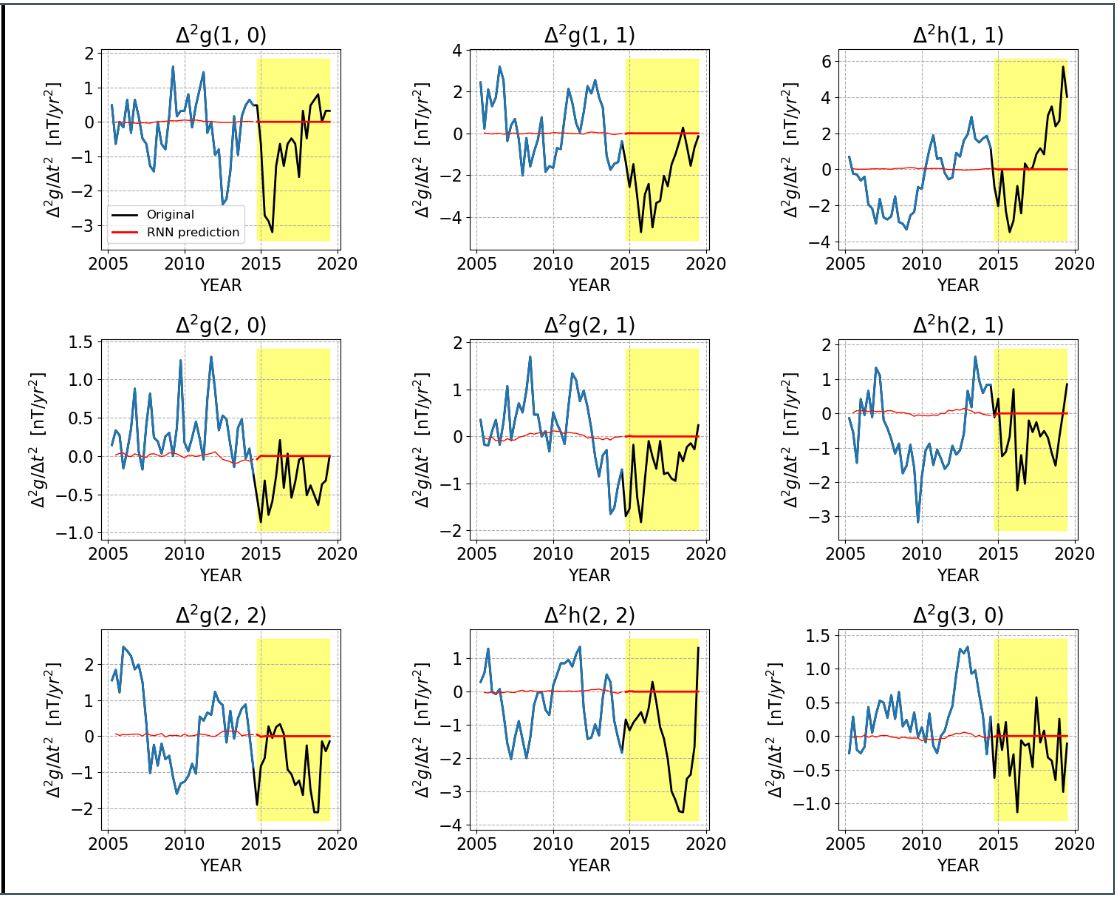
METHOD | Backpropagation algorithm

“What is backpropagation really doing? | Chapter 3, Deep learning” by 3Blue1Brown. (https://youtu.be/Ilg3gGewQ5U?si=SUjGzTFV7ELBlo1v)

METHOD | Overfitting

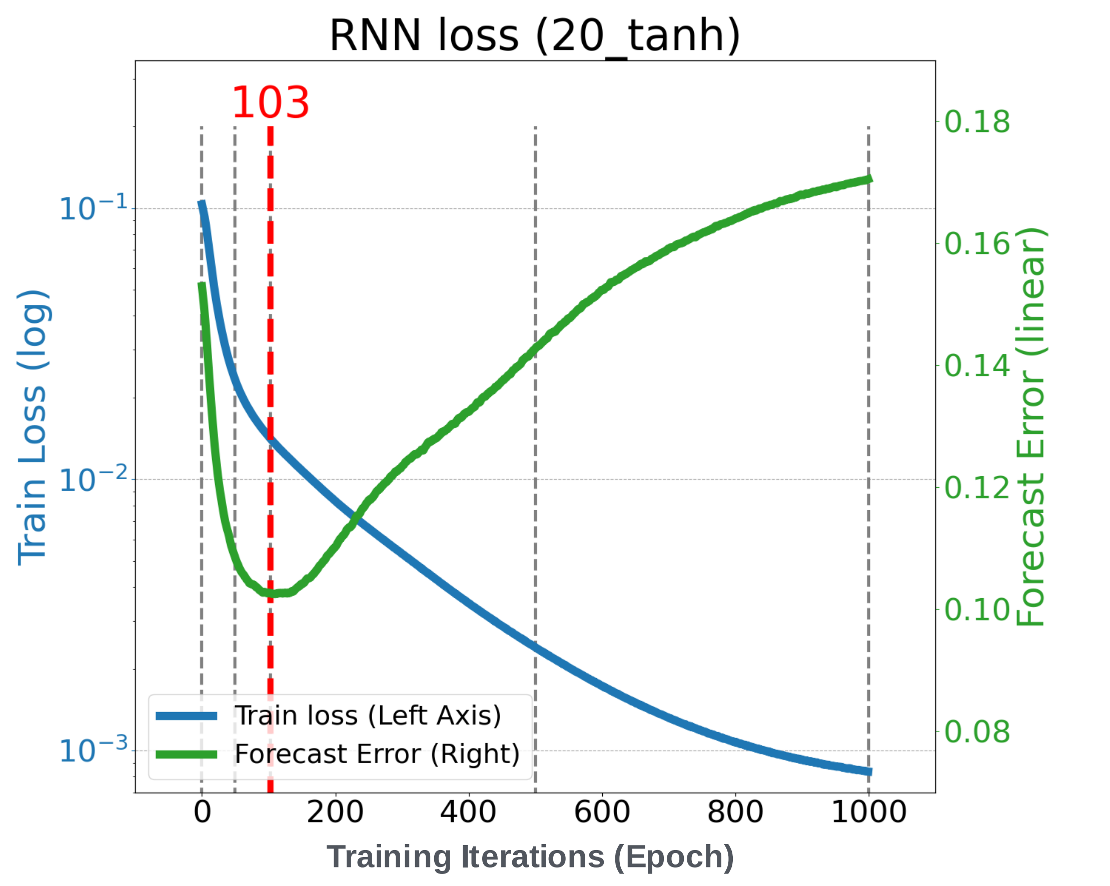
Overfitting caused by too much iteration of the Backpropagation algorithm

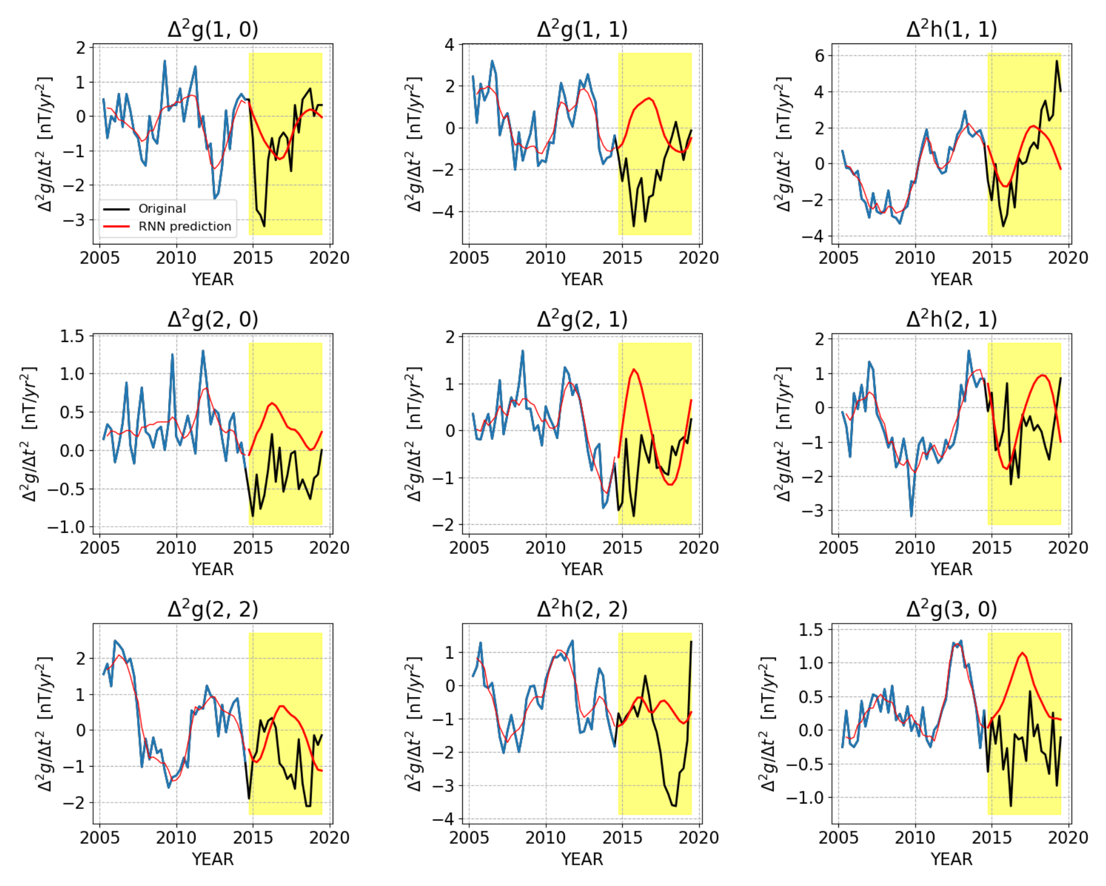




~APR. PROBLEM | Overfitting in RNNs

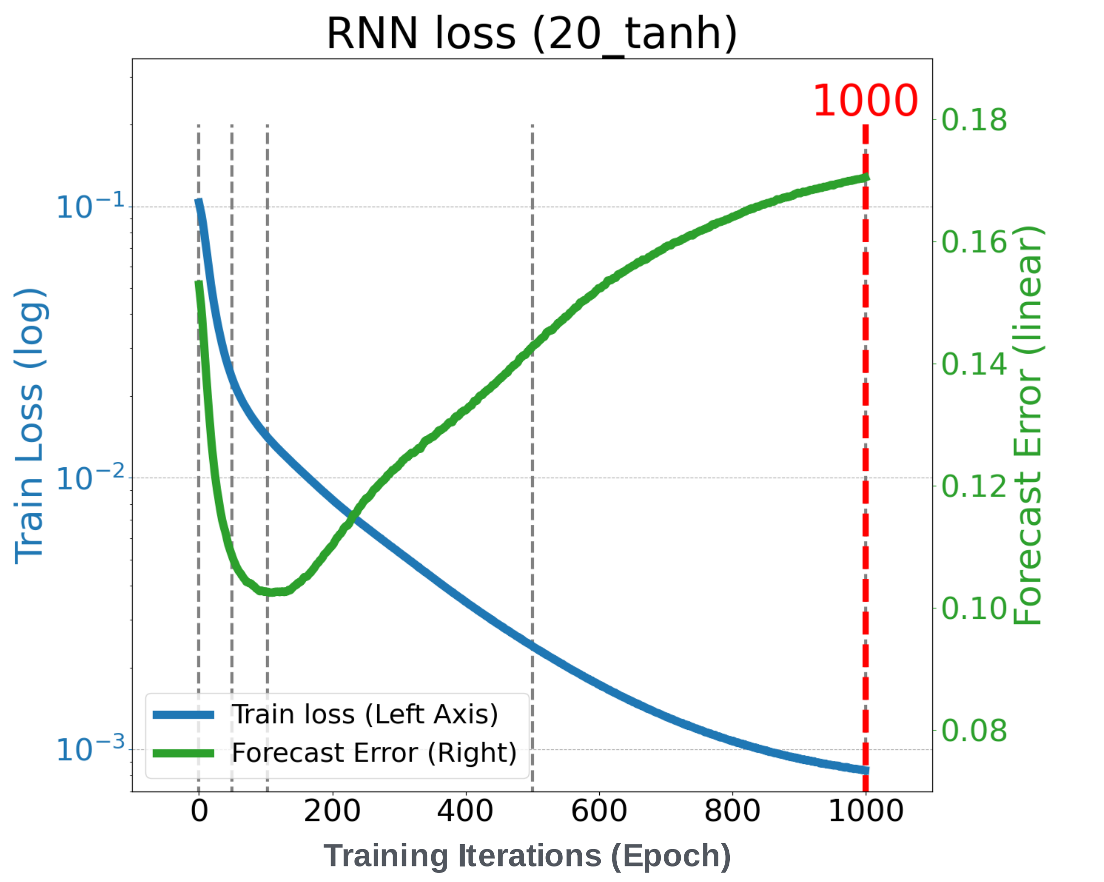
Overfitting caused by too much iteration of the Backpropagation algorithm

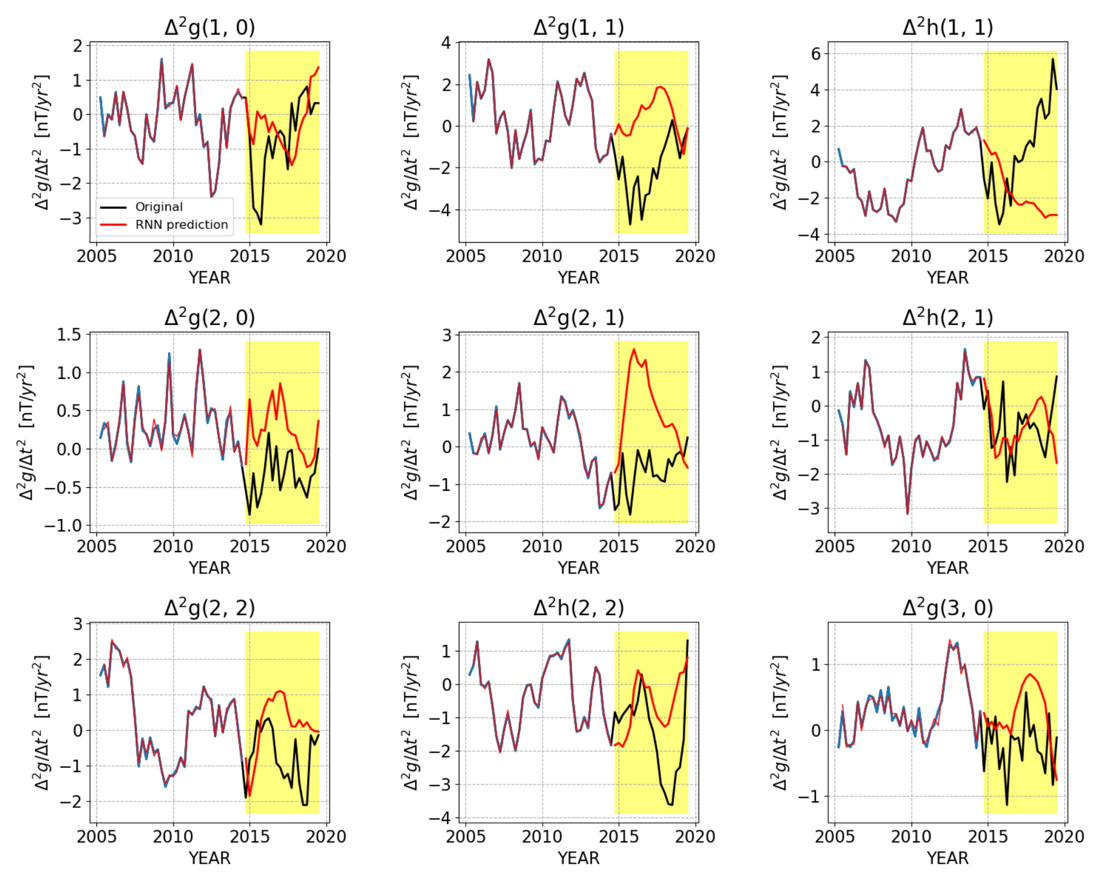




~APR. PROBLEM | Overfitting in RNNs

Overfitting caused by too much iteration of the Backpropagation algorithm





METHOD | EKF algorithm

- State space: $\mathbf{w} = [\, \mathbf{W}^{(\mathrm{in})},\ \mathbf{W}^{(\mathrm{rec})},\ \mathbf{b}^{(\mathrm{rec})},\ \mathbf{W}^{(\mathrm{out})},\ \mathbf{b}^{(\mathrm{out})} \,]^T.$
- Transition model: $\mathbf{M} = \mathbf{I} \Leftrightarrow \mathbf{w}^f(t) = \mathbf{w}^a(t-1)\,,\ \mathbf{P}^f(t) = \mathbf{P}^a(t-1).$
- Observational data: Training data

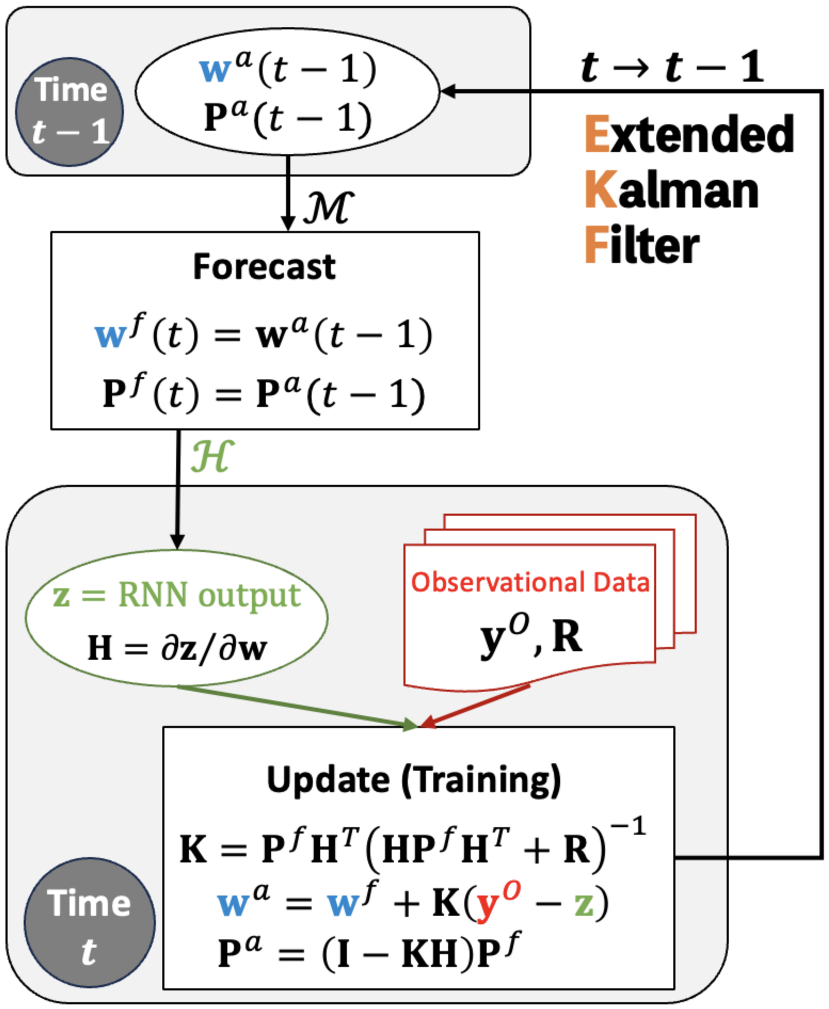
  $\mathbf{y}^O = [\, g_1^0,\ g_1^1,\ h_1^1,\ \cdots,\ g_{13}^{13},\ h_{13}^{13} \,]^T,$ with Observation error - $\mathbf{R}$
- Observation Operator: Forward propagation of RNN

  $\mathcal{H}_t\left(\mathbf{w}^f(t)\right) = \mathrm{RNN}(\,\mathbf{y}^O(t-1) \mid \mathbf{w}^a(t-1)\,) = \mathbf{z}(t).$

  $\mathbf{H}(t) = \left.\frac{\partial \mathbf{z}}{\partial \mathbf{w}}\right|_{\mathbf{w}=\mathbf{w}^f(t)} = \left[\frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(\mathrm{in})}},\ \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(\mathrm{rec})}},\ \cdots,\ \frac{\partial \mathbf{z}}{\partial \mathbf{b}^{(\mathrm{out})}}\right].$
- Update equation: Weight update (= Training)

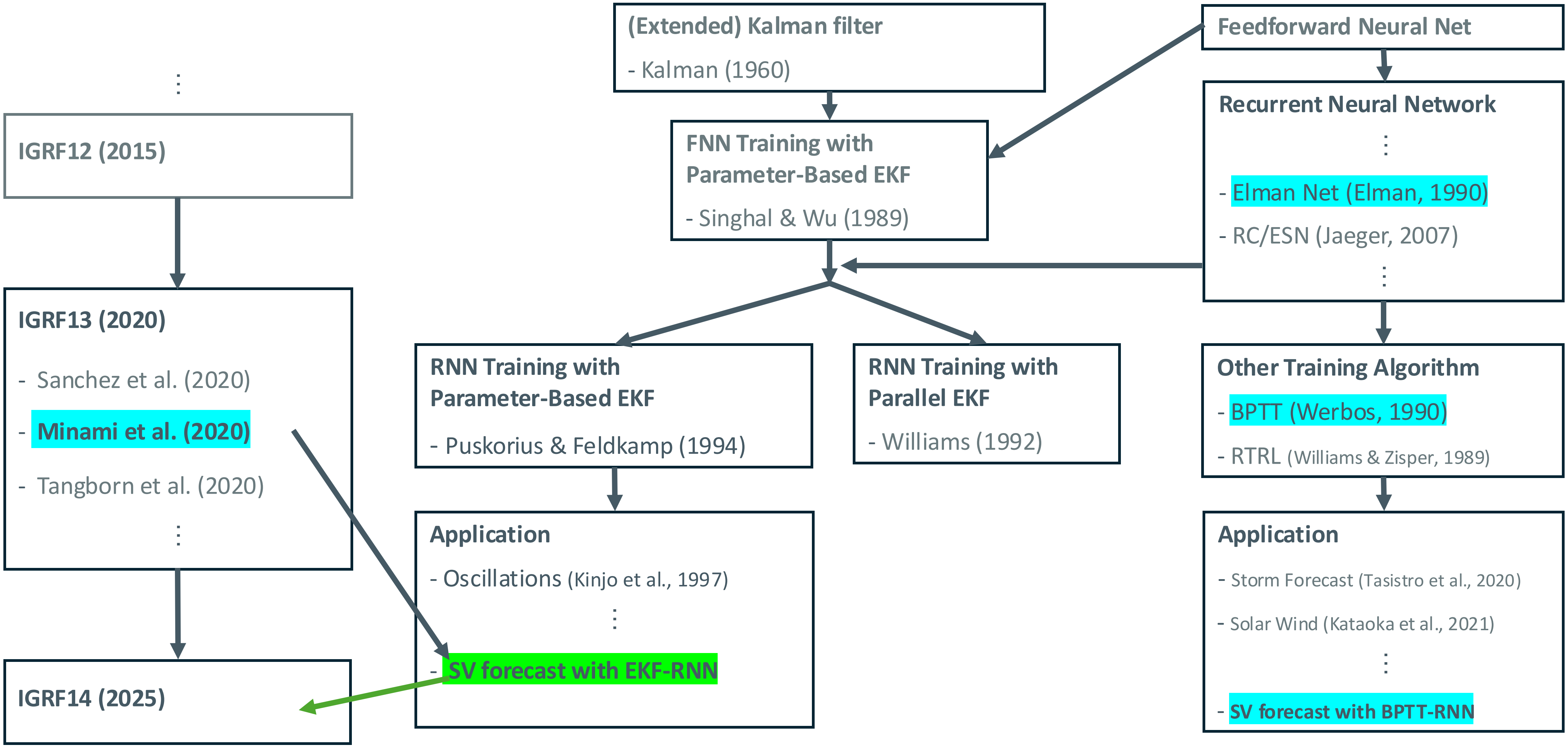
  $\mathbf{w}^a(t) = \mathbf{w}^a(t-1) + \mathbf{K}(t)\left(\mathbf{y}^O(t) - \mathbf{z}(t)\right),\quad \mathbf{P}^a(t) = \left(\mathbf{I} - \mathbf{K}(t)\mathbf{H}(t)\right)\mathbf{P}^f(t).$

Point : ① Forecast error - $\mathbf{P}$. ② Prevents **Overfitting.**



Puskorius and Feldkamp (1994): https://doi.org/10.1109/72.279191.

METHOD | Research Context

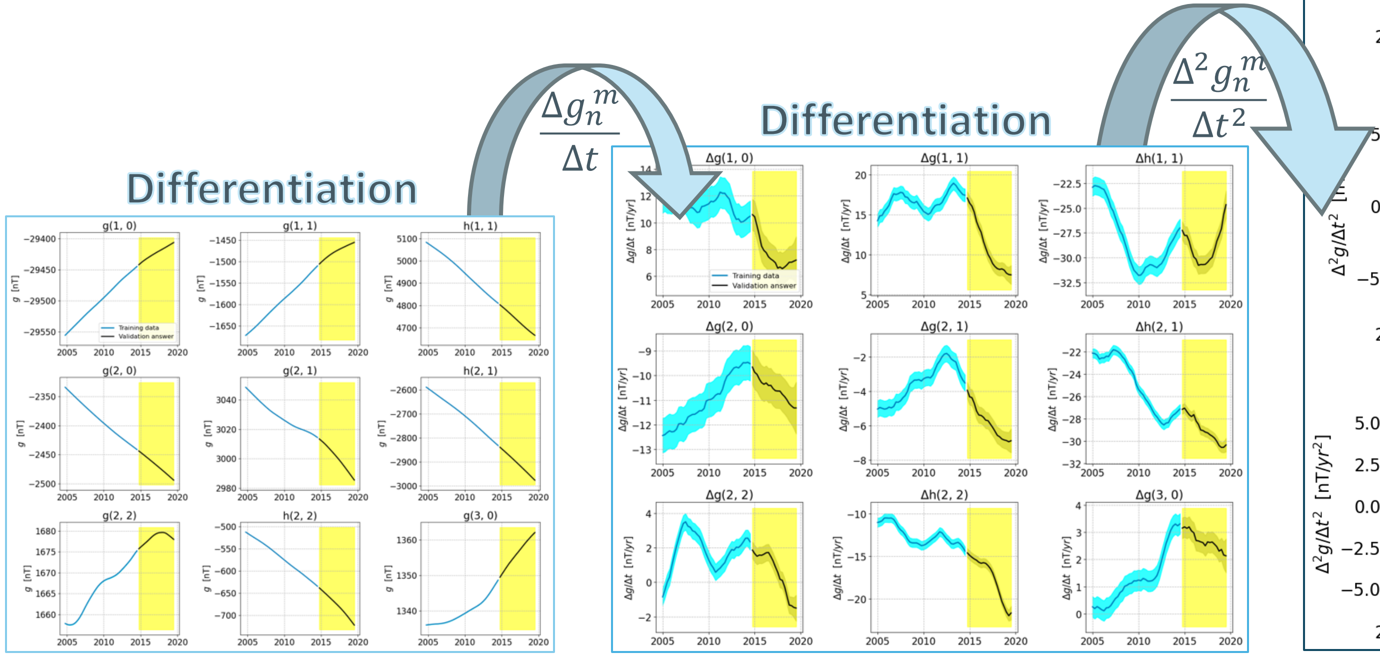


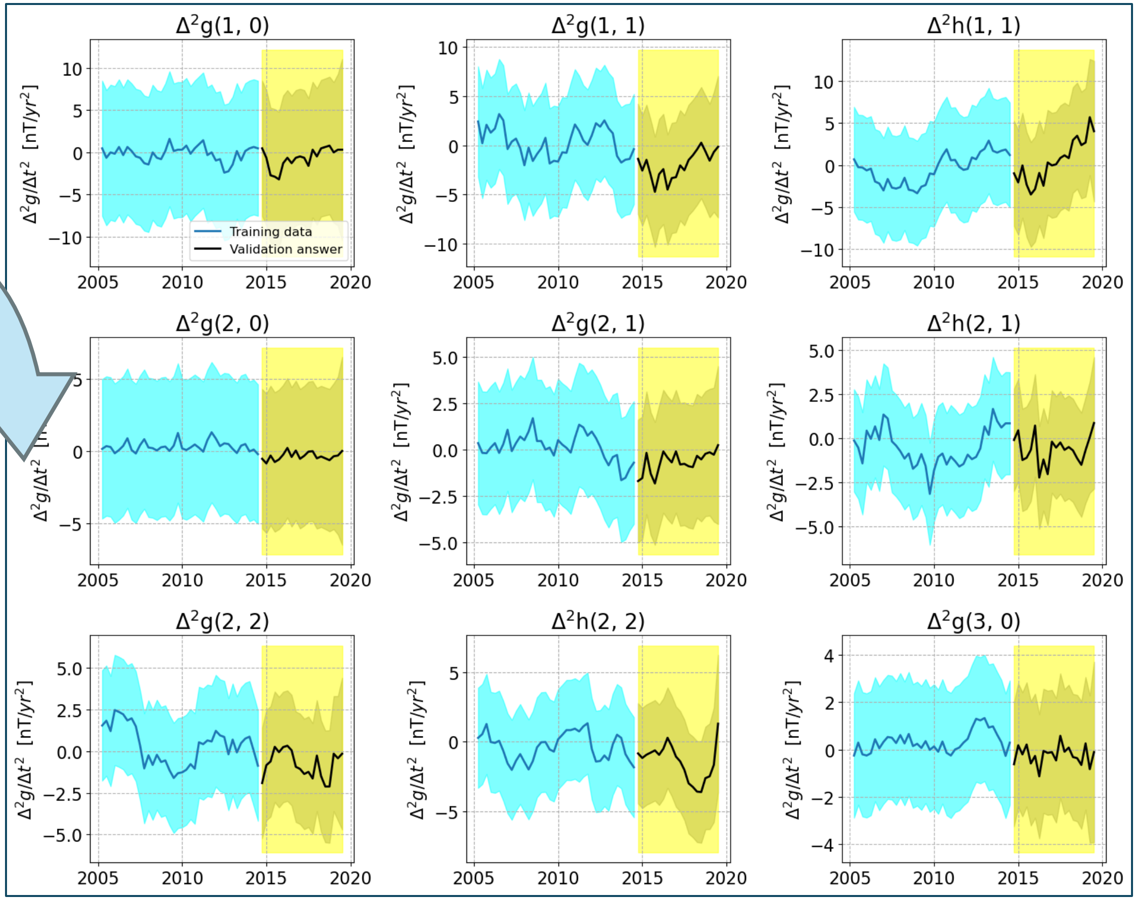
Cf: Jaeger (2002)

METHOD | Hindcast w/ MCM-2020

We assess **RNNs** under the same evaluation setup as in **Minami et al. (2020)**

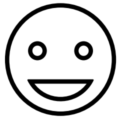
```
    Input Data    : MCM Model (Ropp et al., 2020)
  Training Period : 2004.75 ~ 2014.50 (40 points)
Validation Period : 2014.75 ~ 2019.50 (20 points)
```





- Ropp et al. (2020): https://doi.org/10.1186/s40623-020-01230-1.

HINDCAST | 2nd order time derivative ($\ddot{\mathbf{y}}$)



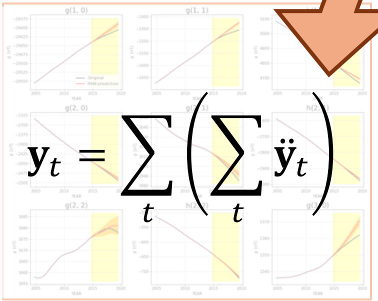
Common Feature:

Predicted dynamics = sin-like wave

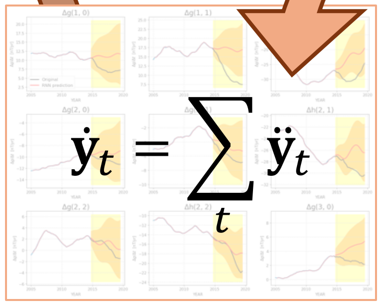
Inaccurate Component (Ex. g_1^0, g_3^0):

Incorrect positive and negative sign

Summation



$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$



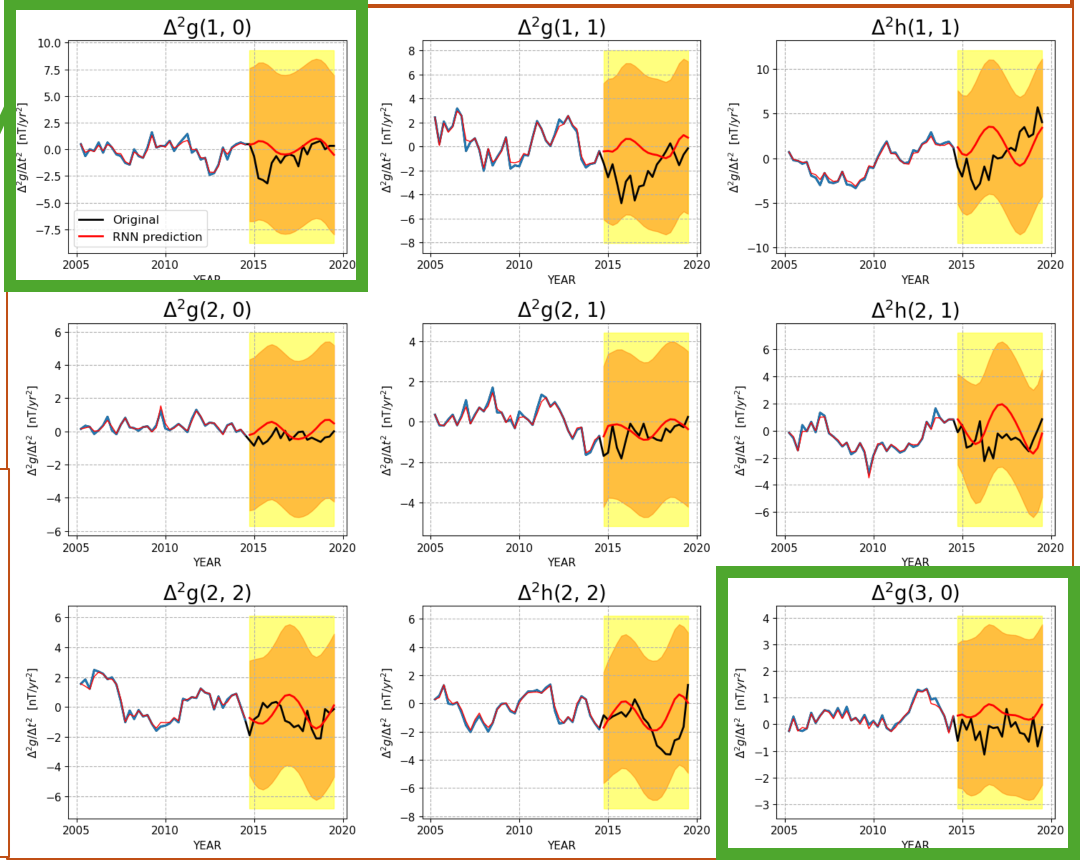
$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$

— Training Data (MCM model)

— Validation Data (MCM model)

— RNN Prediction

$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$



HINDCAST | 1st order time derivative ($\dot{\mathbf{y}}$)

Common Feature:

~~Extends the last trend~~

(Ex. g_2^0, g_3^0)

Inaccurate Component:

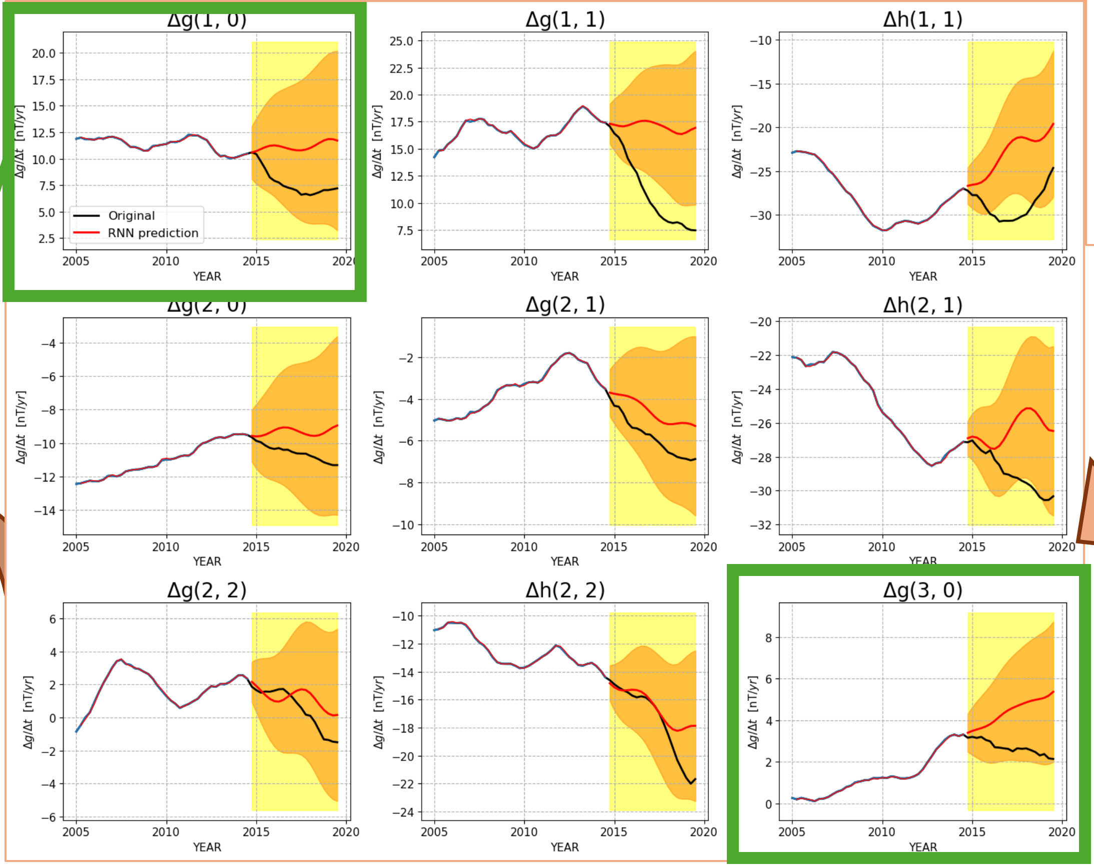
Inaccurate forecast caused by the sudden change of the trend in the validation period (2014).

$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$

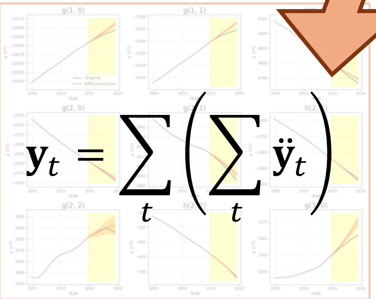
— **Training Data** (MCM model)

— **Validation Data** (MCM model)

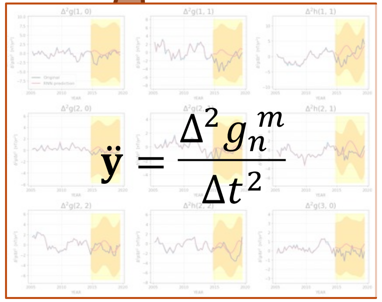
— **RNN Prediction**



Summation



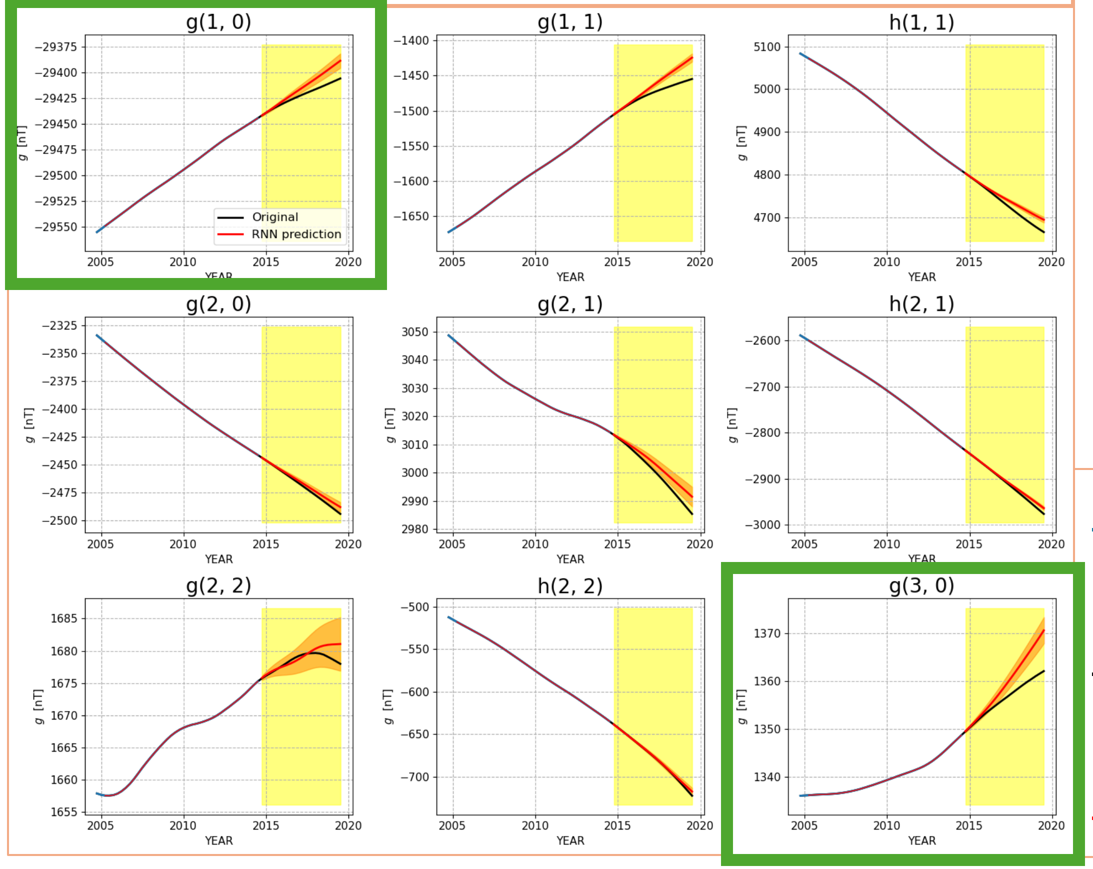
$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$

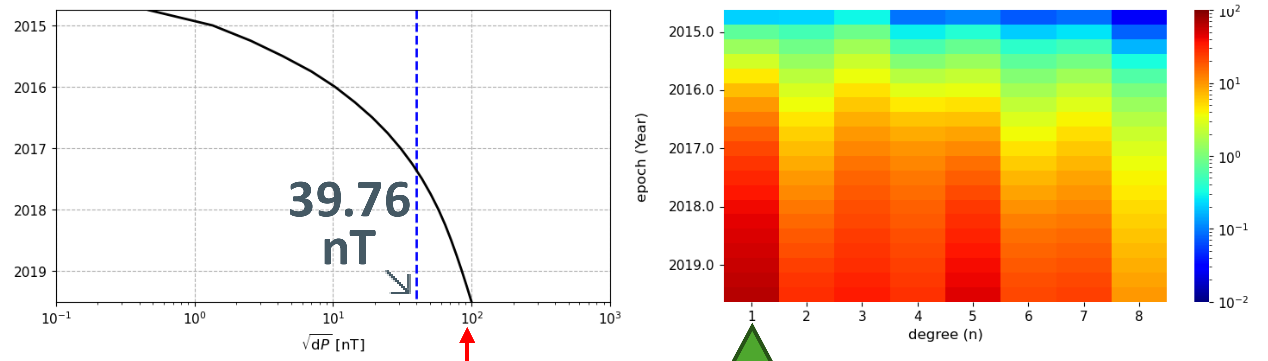


$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

HINDCAST | Original time series (y)

$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$



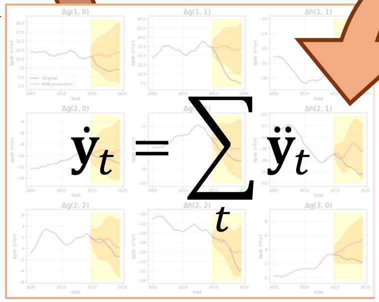


Training Data (MCM model)

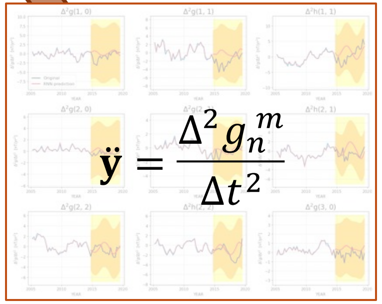
Validation Data (MCM model)

RNN Prediction

Summation



$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$



$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

HINDCAST | Summary Table

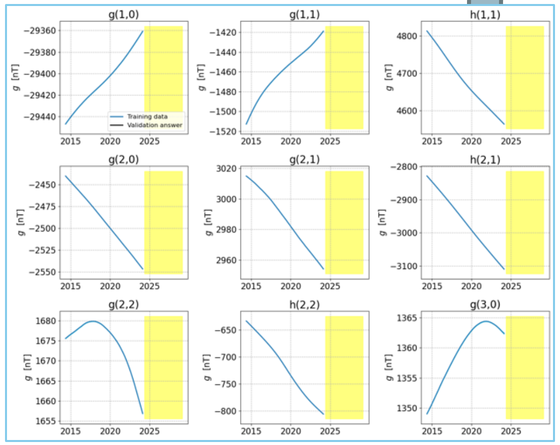
Model	BPTT-RNN	EKF-RNN	Minami et al. (2020) (4D-EnVar)
Average $\sqrt{dP}$	33.43 – 53.69 nT	**37.32 – 46.62 nT**	---
5-year $\sqrt{dP}$	85.83 – 137.74 nT	**99.75 – 117.93 nT**	100.9 – 228.5 nT
Remarks	$\sqrt{dP}$ depends on the # of iteration of BPTT.	$\sqrt{dP}$ depends on the initial $\mathbf{w}_0$.	MHD dynamo (x1000) $\sqrt{dP}$ depends on the assim. window.

Sato & Toh (2024.05; JpGU @Japan)

FORECAST | Forecast w/ MCM-2023

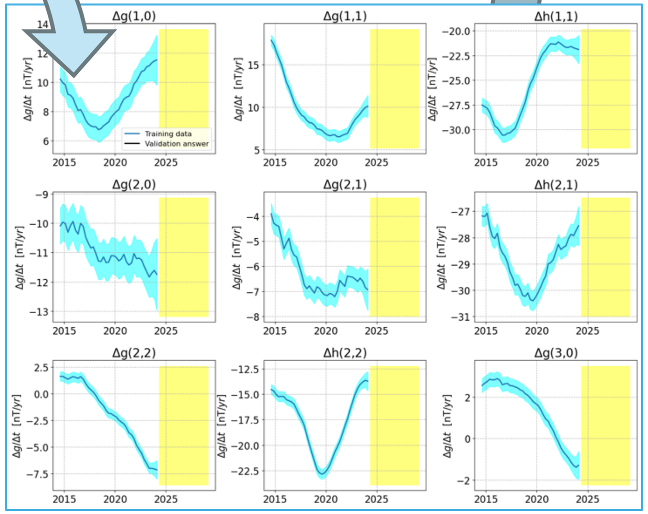
```
    Input Data   : MCM Model (Ropp et al., 2023)
Training Period  : 2014.37 ~ 2024.13 (40 points)
    Test Period  : 2024.37 ~ 2029.13 (20 points)
```

Differentiation

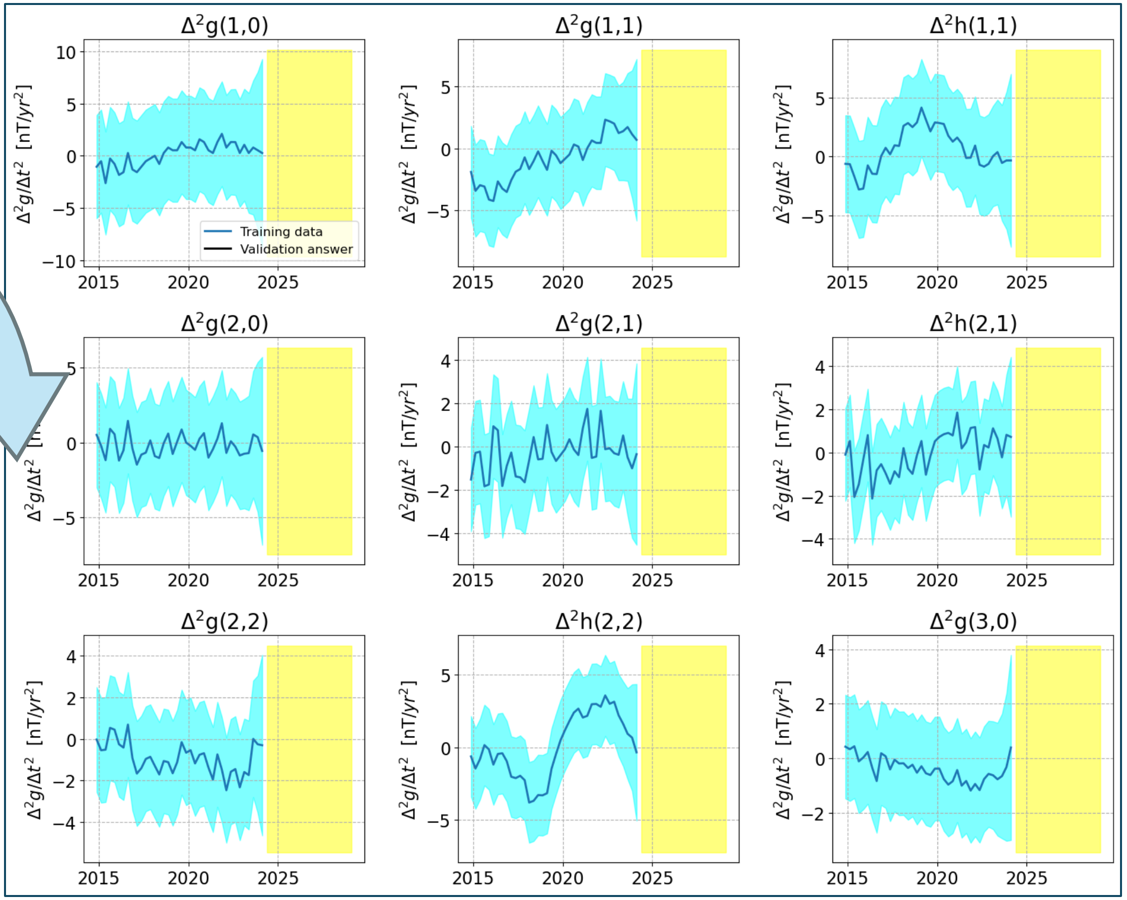


$\frac{\Delta g_n^m}{\Delta t}$

Differentiation



$\frac{\Delta^2 g_n^m}{\Delta t^2}$



Ropp et al. (2023): https://doi.org/10.1093/gji/ggad113.

FORECAST | 2nd order time derivative ($\ddot{\mathbf{y}}$)

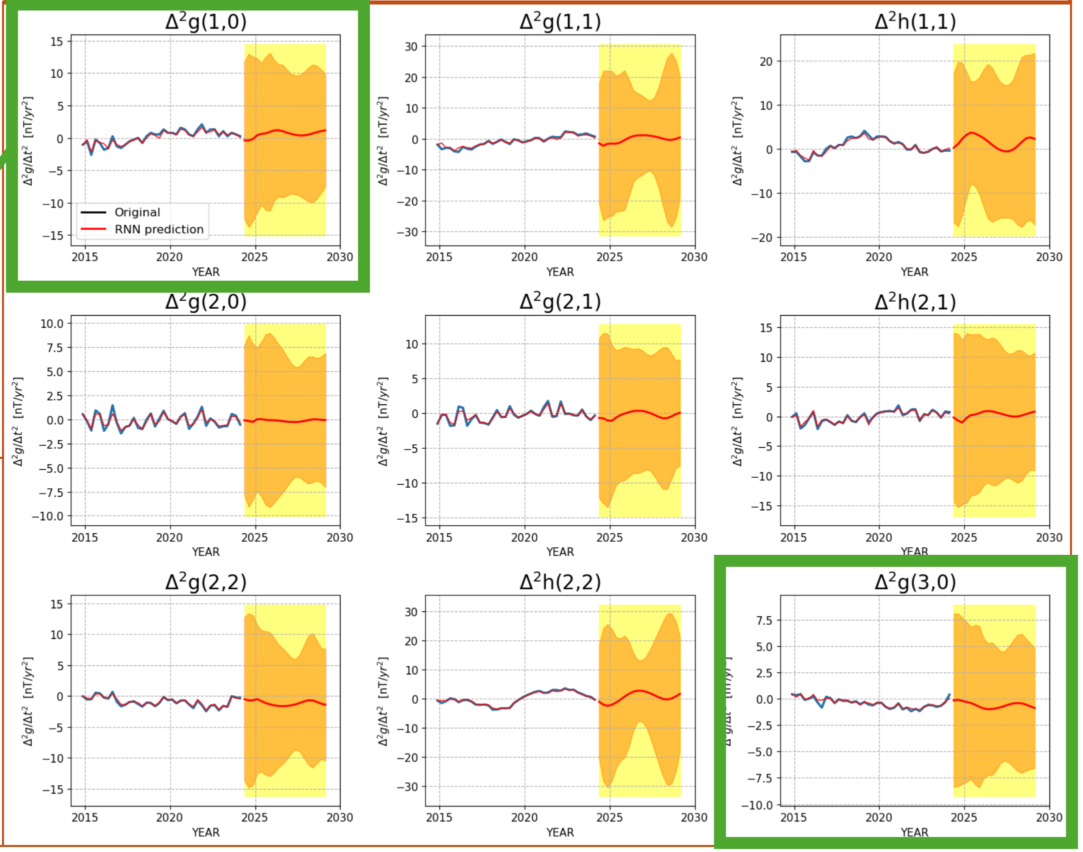
Common Feature:

Predicted dynamics = sin-like wave

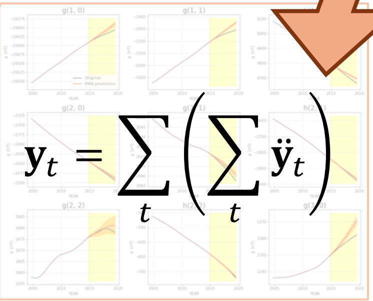
Strong Uncertainty

cf: σ ∽ 5 [nT] in 2015-2020

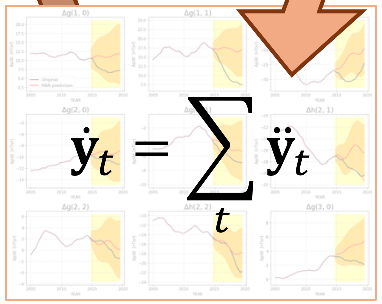
$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$



Summation



$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$



$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$

— **Training Data** (MCM model)

— **Validation Data** (MCM model)

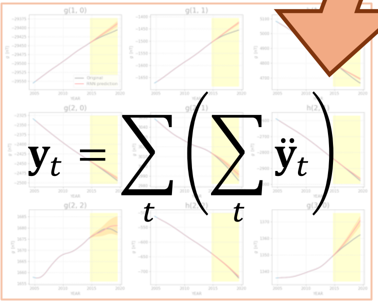
— **RNN Prediction**

FORECAST | 1st order time derivative ($\dot{\mathbf{y}}$)

Common Feature:
~~Extends the last trend~~

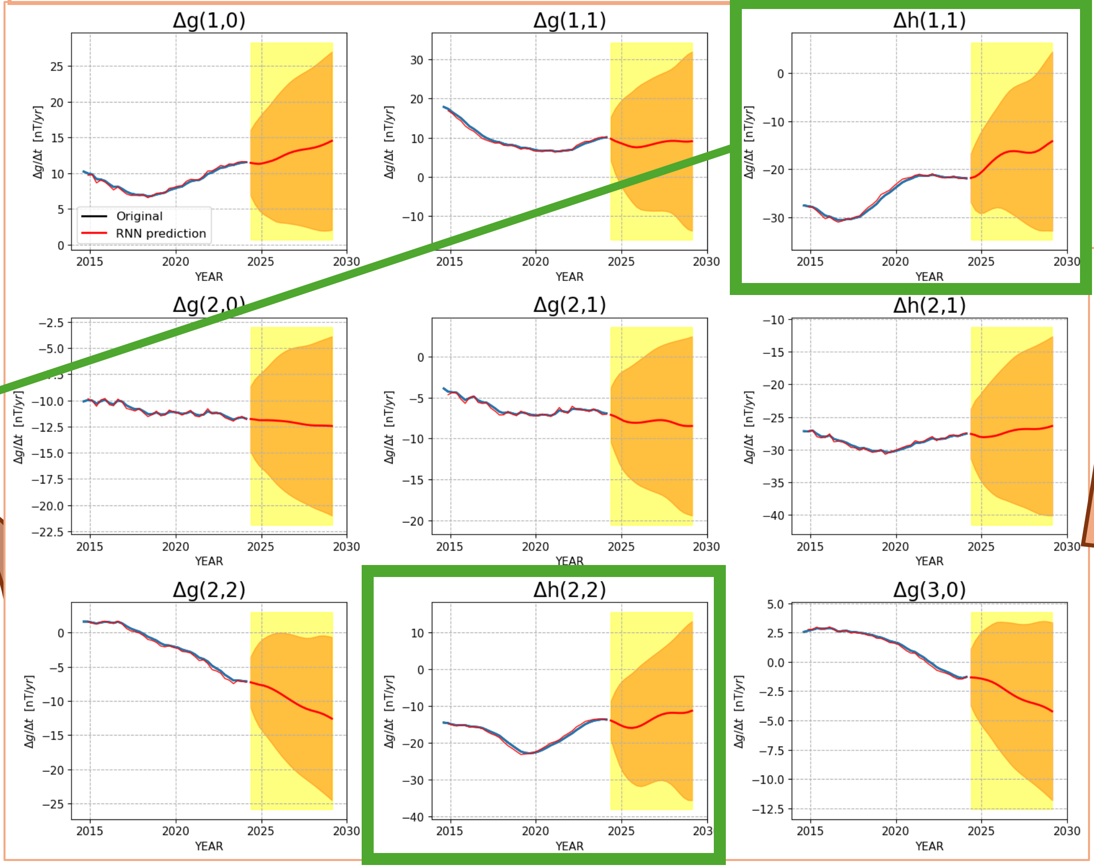
Reproduction of the former dynamics?

Summation

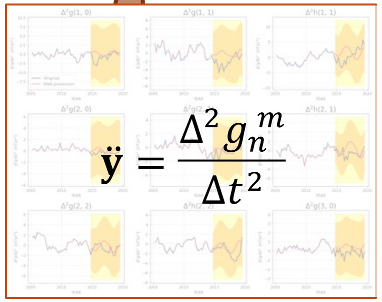


$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$

$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$



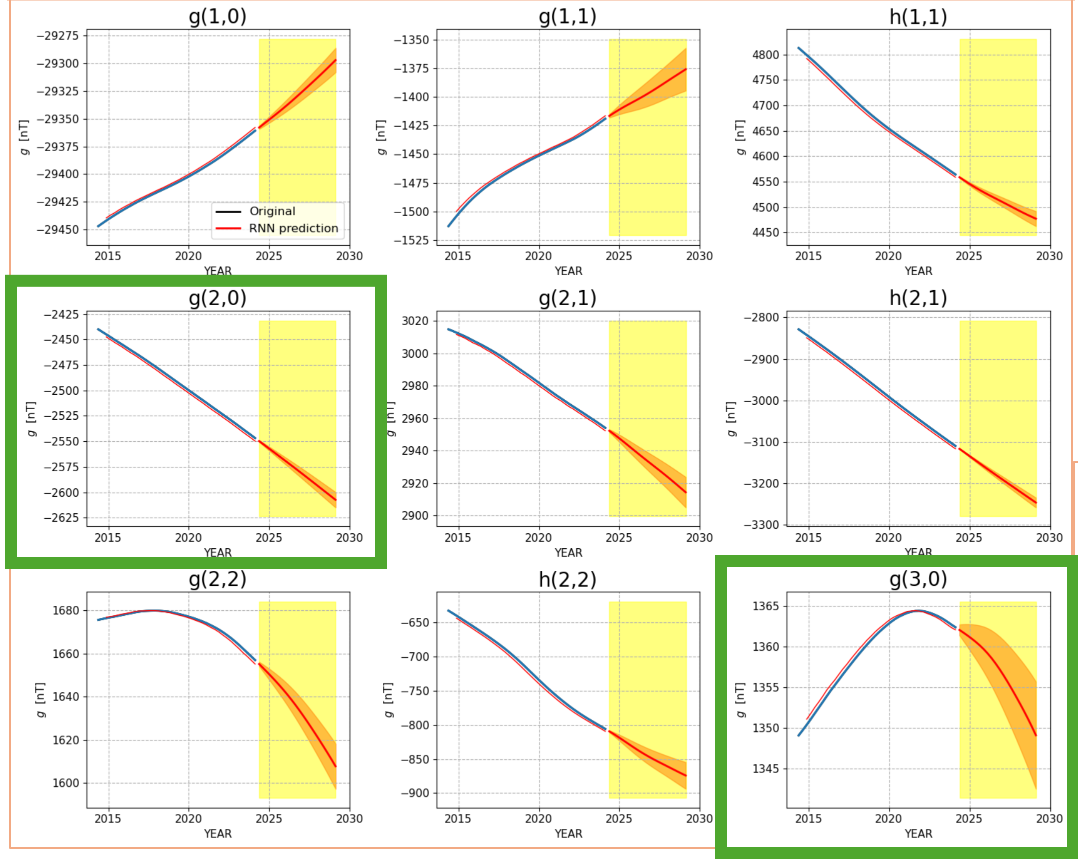
— **Training Data** (MCM model)
— **Validation Data** (MCM model)
— **RNN Prediction**



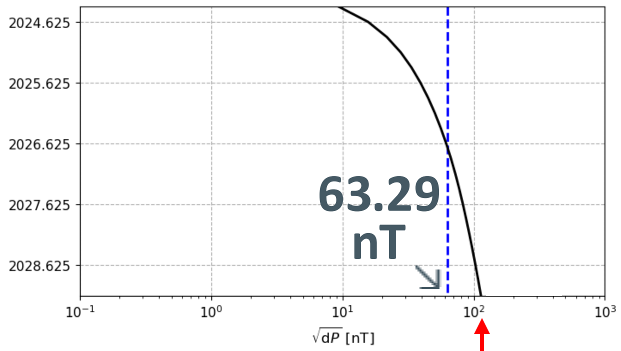
$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

FORECAST | Original time series（$\mathbf{y}$）

$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$

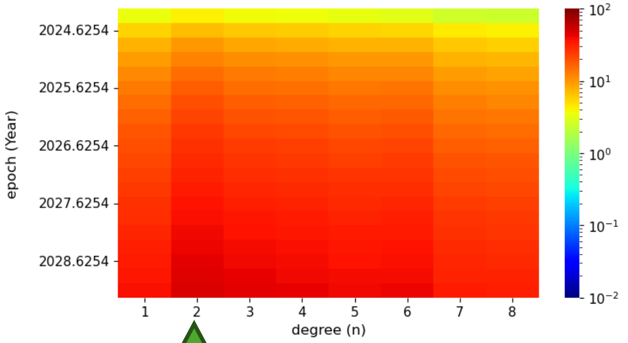


Total Error (estimation)



113.59 nT

Degree dependence (estimation)



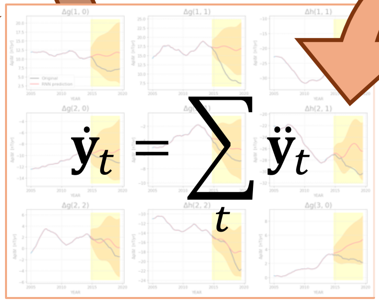
$n = 2$

Training Data (MCM model)

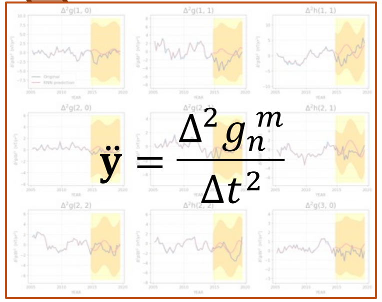
Validation Data (MCM model)

RNN Prediction

Summation



$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$



$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

RESEARCH PLAN | Schedule & Road Map

Short-term forecast of the geomagnetic secular variation using RNNs trained by EKF

Results so far

- EKF-RNN achieves slightly better accuracy than the Geodynamo-based prediction.

ML models can predict geomagnetic variations without geodynamo models.

EKF-RNN uses error covariance matrices of training data, which makes it possible to
① calculate Forecast error - **P**, and ② Prevents **Overfitting**

Schedule & Road Map

- ✅ EKF-RNN x [Geomag **B**] → SV ± Error Hindcast
- ✅ EKF-RNN x [Geomag **B**] → SV ± Error Forecast
- 🔥 Submit pred-model to the IGRF-14 (Deadline=Oct.1)
- ☐ EKF-RNN x [Geomag **B** + Core Surface Flow **u**]

This work is supported by **ROIS-DS-JOINT Grant Number 008RP2024.**

References

- Sanchez, S., Wicht, J. & Bärenzung, J. Predictions of the geomagnetic secular variation based on the ensemble sequential assimilation of geomagnetic field models by dynamo simulations. Earth Planets Space 72, 157 (2020). https://doi.org/10.1186/s40623-020-01279-y.
- Minami, T., Nakano, S., Lesur, V. et al. A candidate secular variation model for IGRF-13 based on MHD dynamo simulation and 4DEnVar data assimilation. Earth Planets Space 72, 136 (2020). https://doi.org/10.1186/s40623-020-01253-8.
- Tangborn, A., Kuang, W., Sabaka, T.J. et al. Geomagnetic secular variation forecast using the NASA GEMS ensemble Kalman filter: A candidate SV model for IGRF-13. Earth Planets Space 73, 47 (2021). https://doi.org/10.1186/s40623-020-01324-w.
- G Ropp, V Lesur, Mid-latitude and equatorial core surface flow variations derived from observatory and satellite magnetic data, Geophysical Journal International, Volume 234, Issue 2, August 2023, Pages 1191–1204, https://doi.org/10.1093/gji/ggad113.
- Ropp, G., Lesur, V., Baerenzung, J. et al. Sequential modelling of the Earth's core magnetic field. Earth Planets Space 72, 153 (2020). https://doi.org/10.1186/s40623-020-01230-1.
- Puskorius GV, Feldkamp LA. Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks. IEEE Trans Neural Netw. 1994;5(2):279-97. https://doi.org/10.1109/72.279191.
- Elman, Jeffrey L. (1990). "Finding Structure in Time". Cognitive Science. 14 (2): 179–211. https://doi.org/10.1207/s15516709cog1402_1.

This work is supported by **ROIS-DS-JOINT Grant Number 008RP2024.**

METHOD | Differentiating training data $\ddot{\mathbf{y}}$ & $\ddot{\mathbf{R}}$

MCM-model: $\mathbf{y}^O = [\, g_0{}^1,\ g_1{}^1,\ h_1{}^1, \ldots, h_{13}^{13}\,]$, $\mathbf{R} =$ Obs-Error Matrix , $(\Delta t = 0.25\text{ year})$

--> 1st deriv: $\dot{\mathbf{y}}^O(t_i) = \dfrac{\mathbf{y}^O(t_i) - \mathbf{y}^O(t_{i-1})}{\Delta t}$, $\dot{\mathbf{R}}(t_i) = \dfrac{1}{\Delta t^2}\big(\mathbf{R}(t_i) + \mathbf{R}(t_{i-1})\big)$

--> 2nd deriv: $\ddot{\mathbf{y}}^O(t_i) = \dfrac{\mathbf{y}^O(t_i) - 2\mathbf{y}^O(t_{i-1}) + \mathbf{y}^O(t_{i-2})}{\Delta t^2}$, $\ddot{\mathbf{R}}(t_i) = \dfrac{1}{\Delta t^4}\big(\mathbf{R}(t_i) + 4\mathbf{R}(t_{i-1}) + \mathbf{R}(t_{i-2})\big)$

P-inflation (to prevent filter divergence): RTPS*1 $\mathbf{P}^{\mathbf{inf}}(t) = \alpha\mathbf{P}^{\mathbf{f}}(t) + (1-\alpha)\mathbf{P}^{\mathbf{a}}(t)$ $(\alpha = 0.5)$

Forecast Loop

2nd deriv: $\mathbf{z}(t_{i+1}) = RNN\big(\mathbf{z}(t_i)\big) = \ddot{\mathbf{y}}^F$, $\mathbf{P}_\mathbf{z}(\mathbf{t}_{i+1}) = \mathbf{G}(t_i)\mathbf{P}_z(t_i)\mathbf{G}(t_i)^T + \mathbf{H}(t_i)\mathbf{P}^a(T)\mathbf{H}(t_i)^T = \ddot{\mathbf{R}}^F$

--> 1st deriv: $\dot{\mathbf{y}}^F(t_{i+1}) = \dot{\mathbf{y}}^F(t_i) + \ddot{\mathbf{y}}^F(t_i)\cdot\Delta t$, $\dot{\mathbf{R}}^F(t_{i+1}) = \dot{\mathbf{R}}^F(t_i) + \ddot{\mathbf{R}}^F(t_i)\cdot\Delta t^2$

--> Target: $\mathbf{y}^F(t_{i+1}) = \mathbf{y}^F(t_i) + \dot{\mathbf{y}}^F(t_i)\cdot\Delta t$, $\mathbf{R}^F(t_{i+1}) = \mathbf{R}^F(t_i) + \dot{\mathbf{R}}^F(t_i)\cdot\Delta t^2 + \ddot{\mathbf{R}}^F(t_i)\cdot\Delta t^4$

$$\left(\mathbf{L}_z(t,\mathbf{x}) = \frac{\partial\mathbf{z}}{\partial\mathbf{x}}\bigg|_{w=\mathbf{w}(t)},\quad \mathbf{L}_w(t,\mathbf{w}) = \frac{\partial\mathbf{z}}{\partial\mathbf{w}}\bigg|_{x=\mathrm{x}(t)},\quad \mathbf{G}(t) = \mathbf{L}_z\big(t,\mathbf{x}(t)\big),\ \ \mathbf{H}(t) = \mathbf{L}_w\big(t,\mathbf{w}(t)\big)\right)$$

*1: Relaxation-to-Prior-Spread method (Whitaker and Hamill, 2012)

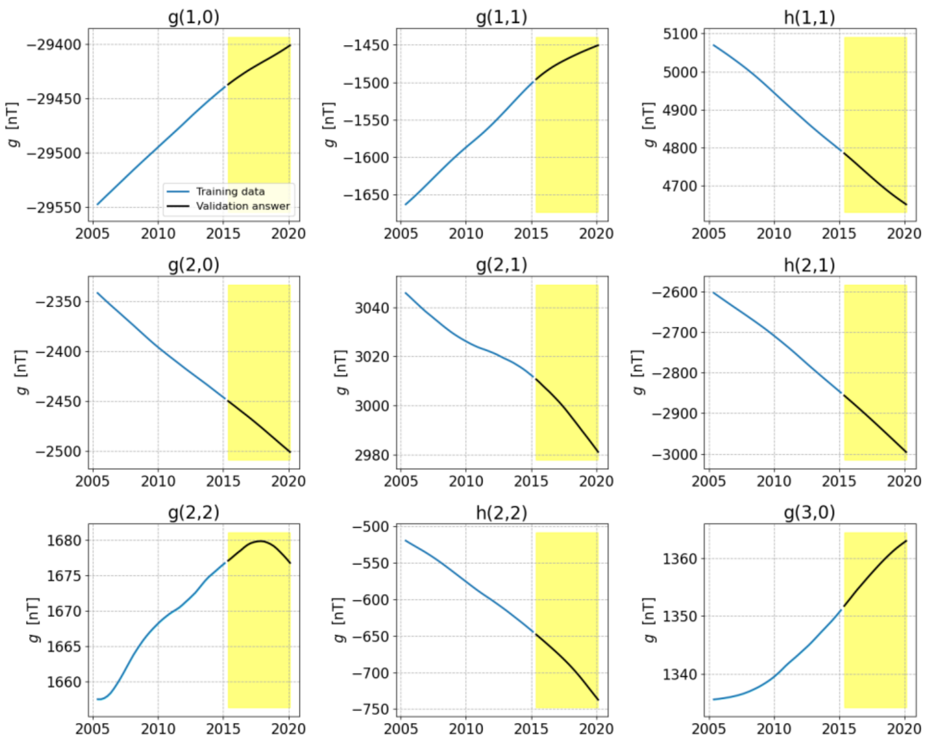
DETAIL | Differentiating training data

$$\mathbf{y}^O(t_i) = [\, g_1^0(t_i), g_1^1(t_i), h_1^1(t_i), \cdots, g_{13}^{13}(t_i), h_{13}^{13}(t_i) \,].$$

$$\boldsymbol{\varepsilon}^O(t_i) = \mathbf{y}^O(t_i) - \mathbf{x}^{\text{true}}(t_i).$$

$$\mathbf{R}(t_i) = \langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle$$

$$= \begin{bmatrix} \delta g_1^0(t_i)^2 & \cdots & \langle \text{Cross Term} \rangle \\ \vdots & \ddots & \vdots \\ \langle \text{Cross Term} \rangle & \cdots & \delta h_{13}^{13}(t_i)^2 \end{bmatrix}.$$



DETAIL | Differentiating training data

$$\dot{\mathbf{y}}^O(t_i) = \frac{\mathbf{y}^O(t_i) - \mathbf{y}^O(t_{i-1})}{\Delta t}.$$

$$\left(\begin{array}{c}\text{Backward diff.}\\ \Delta t = 0.25\ \text{year}\end{array}\right)$$

$$\dot{\mathbf{x}}^{\text{true}}(t_i) = \frac{\mathbf{x}^{\text{true}}(t_i) - \mathbf{x}^{\text{true}}(t_{i-1})}{\Delta t}.$$

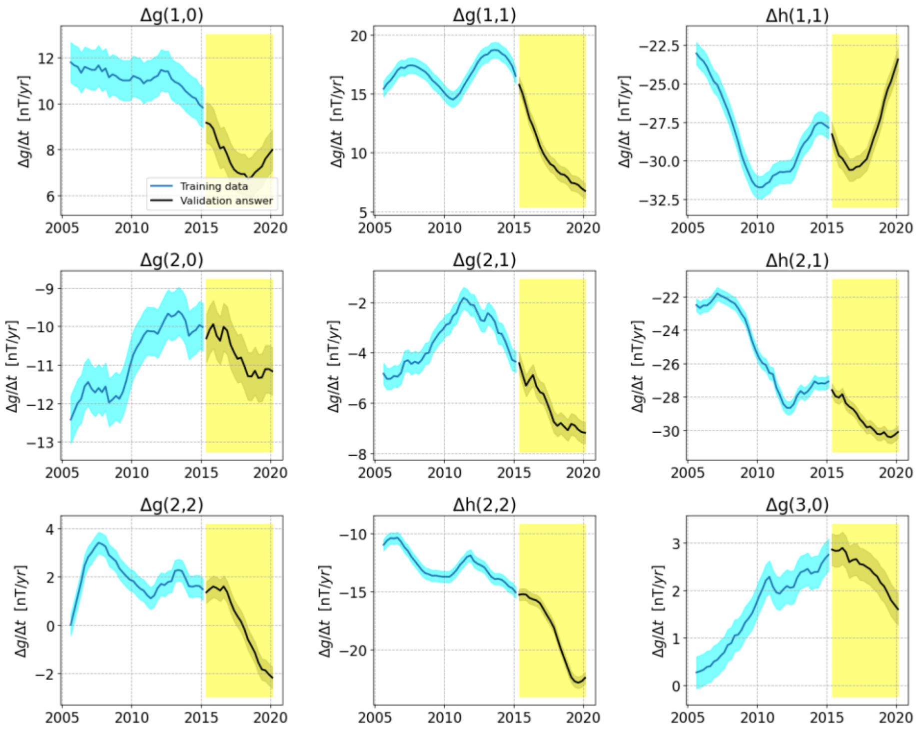
$$\dot{\boldsymbol{\varepsilon}}^O(t_i) = \dot{\mathbf{y}}^O(t_i) - \dot{\mathbf{x}}^{\text{true}}(t_i) = \frac{1}{\Delta t}\left(\boldsymbol{\varepsilon}^O(t_i) - \boldsymbol{\varepsilon}^O(t_{i-1})\right).$$

$$\dot{\mathbf{R}}(t_i) = \langle \dot{\boldsymbol{\varepsilon}}^O(t_i) \cdot \dot{\boldsymbol{\varepsilon}}^O(t_i)^T \rangle$$

$$= \frac{1}{\Delta t^2}\left(\begin{array}{c}\langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle + \langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle \\ + \langle \boldsymbol{\varepsilon}^O(t_{i-1}) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle + \langle \boldsymbol{\varepsilon}^O(t_{i-1}) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle\end{array}\right).$$

Assume $\langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle = \langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle = 0,$

$$\dot{\mathbf{R}}(t_i) = \mathbf{R}^O(t_i) + \mathbf{R}^O(t_{i-1})$$



DETAIL | Differentiating training data

$$\ddot{\mathbf{y}}^O(t_i) = \frac{\dot{\mathbf{y}}^O(t_i) - \dot{\mathbf{y}}^O(t_{i-1})}{\Delta t} = \frac{\mathbf{y}^O(t_i) - 2\mathbf{y}^O(t_{i-1}) + \mathbf{y}^O(t_{i-2})}{\Delta t^2}.$$

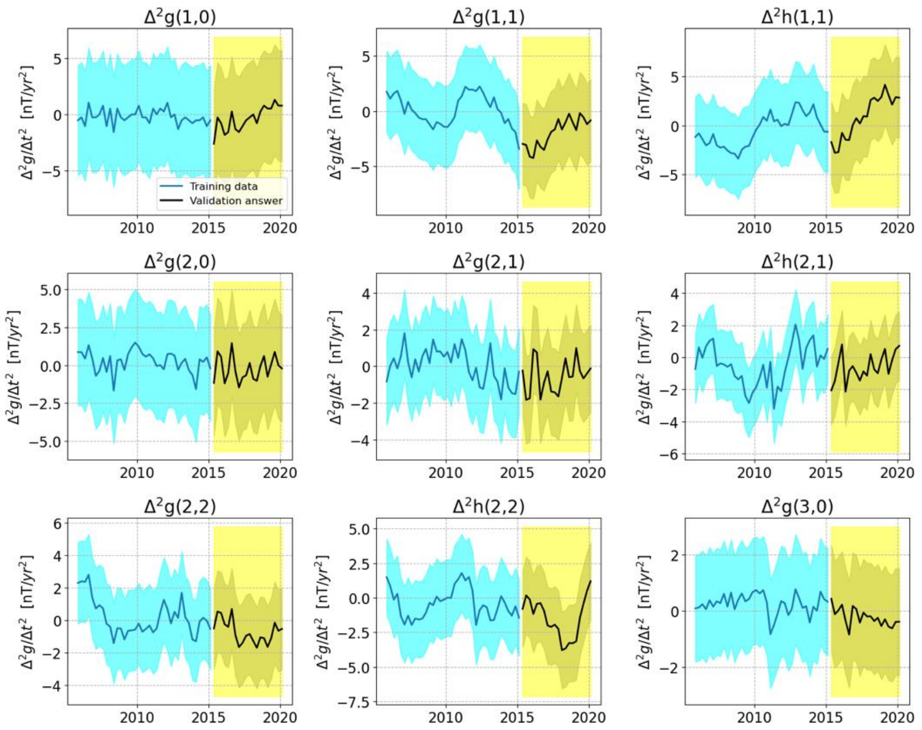
$$\ddot{\boldsymbol{\varepsilon}}^O(t_i) = \ddot{\mathbf{y}}^O(t_i) - \ddot{\mathbf{x}}^{\mathrm{true}}(t_i) = \frac{1}{\Delta t^2}\left(\boldsymbol{\varepsilon}^O(t_i) - 2\boldsymbol{\varepsilon}^O(t_{i-1}) + \boldsymbol{\varepsilon}^O(t_{i-2})\right).$$

$$\ddot{\mathbf{R}}(t_i) = \langle \ddot{\boldsymbol{\varepsilon}}^O(t_i) \cdot \ddot{\boldsymbol{\varepsilon}}^O(t_i)^T \rangle$$

$$= \frac{1}{\Delta t^4}\begin{pmatrix} \langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle - 2\langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle + \langle \boldsymbol{\varepsilon}^O(t_i) \cdot \boldsymbol{\varepsilon}^O(t_{i-2})^T \rangle \\ -2\langle \boldsymbol{\varepsilon}^O(t_{i-1}) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle + 4\langle \boldsymbol{\varepsilon}^O(t_{i-1}) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle - 2\langle \boldsymbol{\varepsilon}^O(t_{i-1}) \cdot \boldsymbol{\varepsilon}^O(t_{i-2})^T \rangle \\ +\langle \boldsymbol{\varepsilon}^O(t_{i-2}) \cdot \boldsymbol{\varepsilon}^O(t_i)^T \rangle - 2\langle \boldsymbol{\varepsilon}^O(t_{i-2}) \cdot \boldsymbol{\varepsilon}^O(t_{i-1})^T \rangle + \langle \boldsymbol{\varepsilon}^O(t_{i-2}) \cdot \boldsymbol{\varepsilon}^O(t_{i-2})^T \rangle \end{pmatrix}.$$

Assume $\langle \boldsymbol{\varepsilon}^O(t_j) \cdot \boldsymbol{\varepsilon}^O(t_k)^T \rangle = \begin{cases} \left\langle \boldsymbol{\varepsilon}^O(t_j)^2 \right\rangle & (\text{for } j = k) \\ 0 & (\text{for } j \neq k) \end{cases},$

$$\dot{\mathbf{R}}(t_i) = \mathbf{R}^O(t_i) + 4\mathbf{R}^O(t_{i-1}) + \mathbf{R}^O(t_{i-2})$$



DETAIL | Integrating forecast data

$$\ddot{\mathbf{y}}^F(t_i) = \mathrm{RNN}\left(\ddot{\mathbf{y}}^F(t_{i-1})\right), \qquad \mathbf{P}_\mathbf{z}(t_i) = \mathbf{G}(t_{i-1})\mathbf{P}_z(t_{i-1})\mathbf{G}(t_{i-1})^T + \mathbf{H}(t_{i-1})\mathbf{P}^a(t_{\mathrm{Rel}})\mathbf{H}(t_{i-1})^T = \ddot{\mathbf{R}}^F(t_i).$$

$$\begin{aligned}
\dot{\mathbf{y}}^F(t_i) &= \dot{\mathbf{y}}^F(t_{i-1}) + \ddot{\mathbf{y}}^F(t_i)\cdot\Delta t \\
\dot{\mathbf{x}}^{\mathrm{true}}(t_i) &= \dot{\mathbf{x}}^{\mathrm{true}}(t_{i-1}) + \ddot{\mathbf{x}}^{\mathrm{true}}(t_i)\cdot\Delta t
\end{aligned}$$

$$\dot{\boldsymbol{\varepsilon}}^F(t_i) = \dot{\boldsymbol{\varepsilon}}^F(t_{i-1}) + \ddot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\Delta t$$

Assume $\langle\dot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\ddot{\boldsymbol{\varepsilon}}^F(t_i)^T\rangle = \langle\ddot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\dot{\boldsymbol{\varepsilon}}^F(t_i)^T\rangle = 0$

$$\dot{\mathbf{R}}^F(t_i) = \dot{\mathbf{R}}^F(t_{i-1}) + \ddot{\mathbf{R}}^F(t_{i-1})\cdot\Delta t^2$$

$$\begin{aligned}
\mathbf{y}^F(t_i) &= \mathbf{y}^F(t_{i-1}) + \dot{\mathbf{y}}^F(t_i)\cdot\Delta t \\
\mathbf{x}^{\mathrm{true}}(t_i) &= \mathbf{x}^{\mathrm{true}}(t_{i-1}) + \dot{\mathbf{x}}^{\mathrm{true}}(t_i)\cdot\Delta t
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{\varepsilon}^F(t_i) &= \boldsymbol{\varepsilon}^F(t_{i-1}) + \dot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\Delta t \\
&= \boldsymbol{\varepsilon}^F(t_{i-1}) + \dot{\boldsymbol{\varepsilon}}^F(t_{i-1})\cdot\Delta t + \ddot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\Delta t^2
\end{aligned}$$

Assume $\langle\boldsymbol{\varepsilon}^F(t_i)\cdot\dot{\boldsymbol{\varepsilon}}^F(t_i)^T\rangle = \langle\dot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\boldsymbol{\varepsilon}^F(t_i)^T\rangle$

$\langle\boldsymbol{\varepsilon}^F(t_i)\cdot\ddot{\boldsymbol{\varepsilon}}^F(t_i)^T\rangle = \langle\ddot{\boldsymbol{\varepsilon}}^F(t_i)\cdot\boldsymbol{\varepsilon}^F(t_i)^T\rangle = 0$

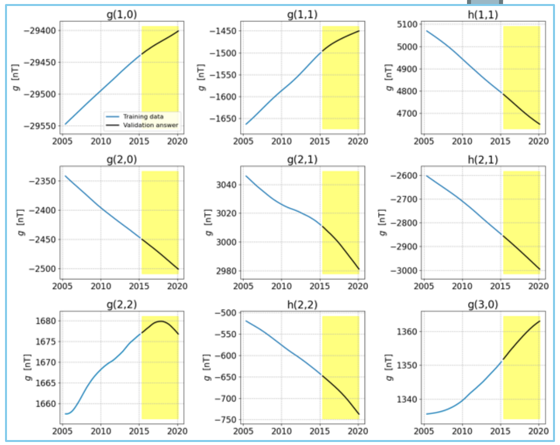
$$\mathbf{R}^F(t_i) = \mathbf{R}^F(t_{i-1}) + \dot{\mathbf{R}}^F(t_{i-1})\cdot\Delta t^2 + \ddot{\mathbf{R}}^F(t_{i-1})\cdot\Delta t^4$$

METHOD | Hindcast w/ MCM-2023

We assess **RNNs** under ~~the same evaluation setup as in **Minami et al. (2020)**~~

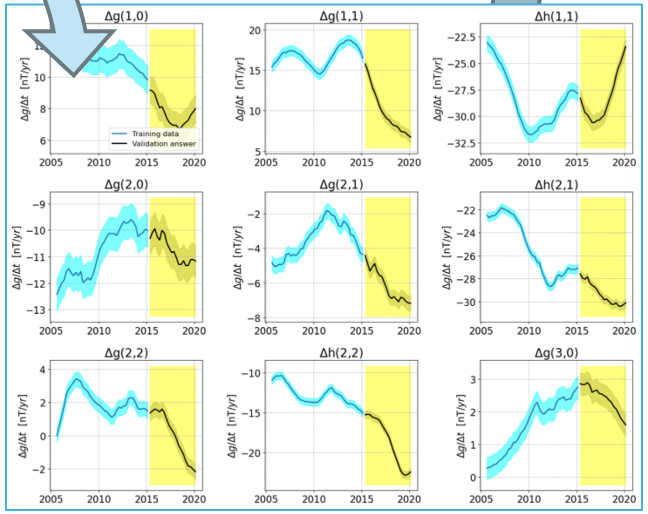
Input Data : MCM Model (Ropp et al., 2023)
Training Period : 2005.37 ~ 2015.13 (40 points)
Validation Period : 2015.63 ~ 2020.13 (20 points)

Differentiation

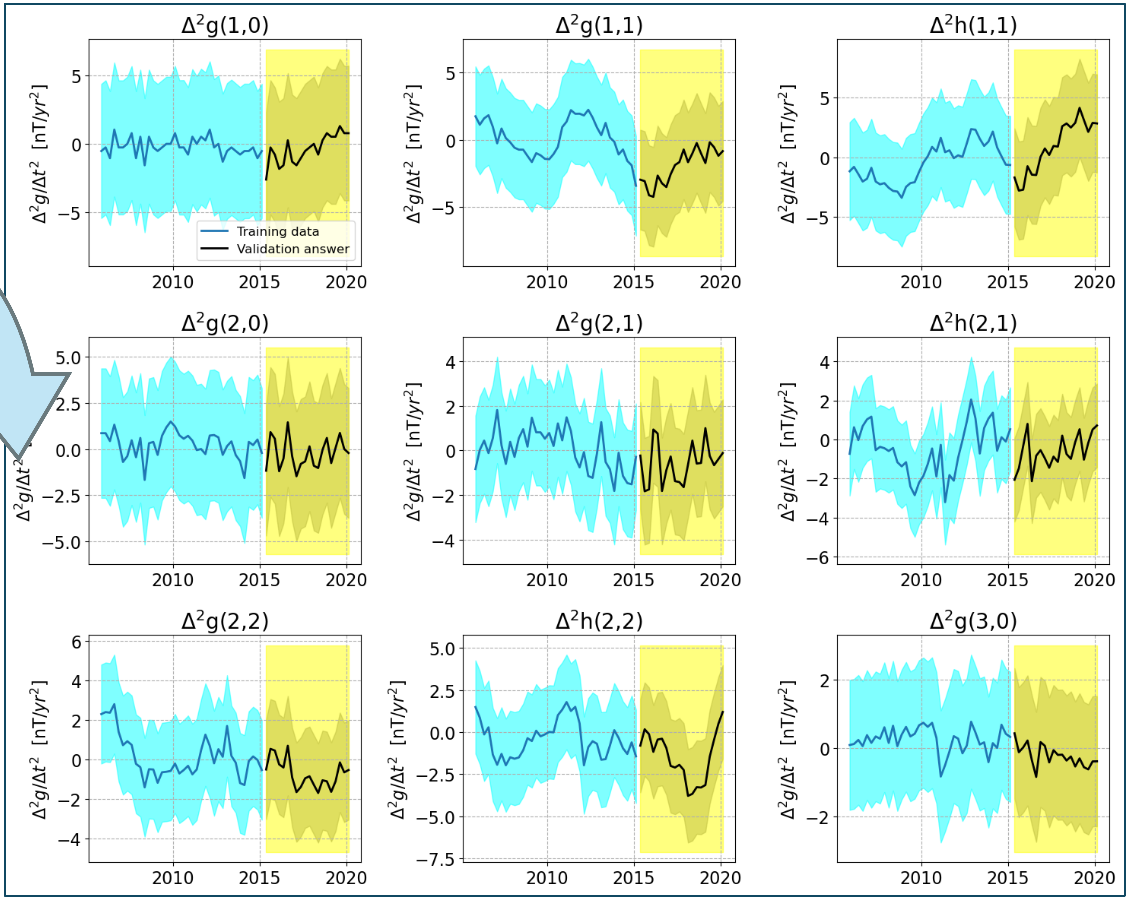


$\frac{\Delta g_n^m}{\Delta t}$

Differentiation



$\frac{\Delta^2 g_n^m}{\Delta t^2}$



Ropp et al. (2023): https://doi.org/10.1093/gji/ggad113.

HINDCAST | 2nd order time derivative ($\ddot{\mathbf{y}}$)

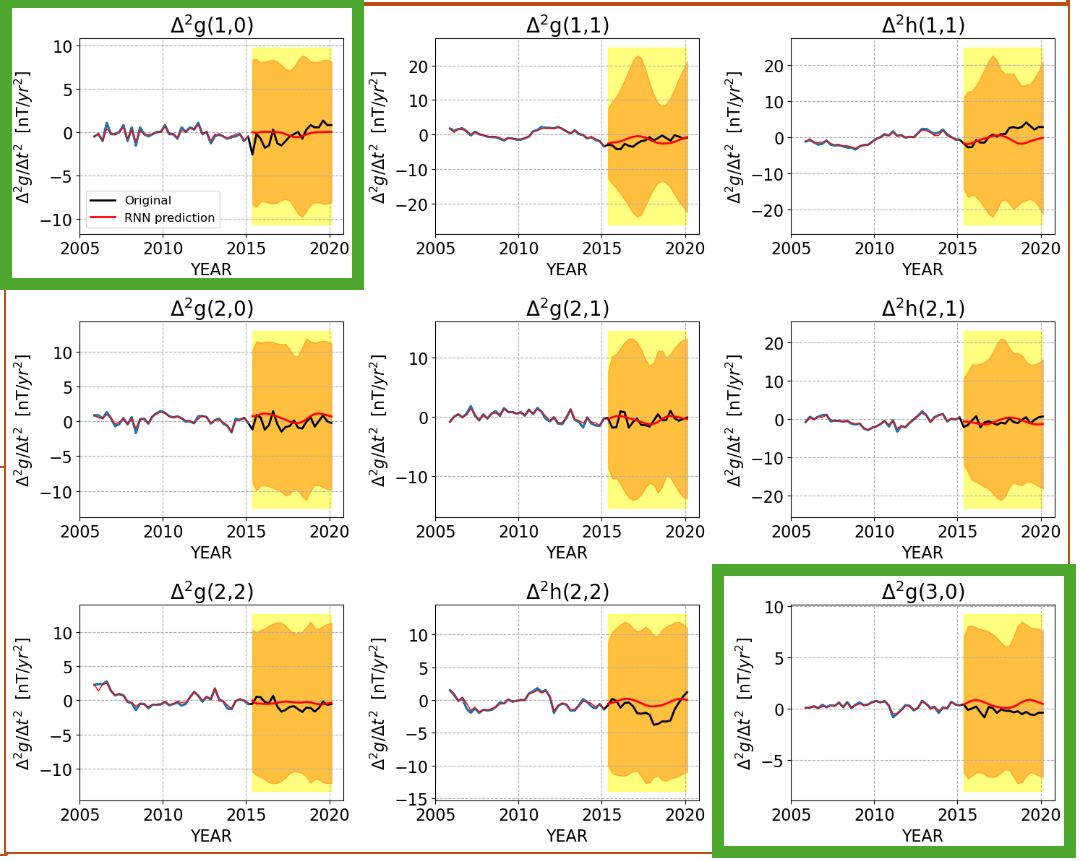
Common Feature:

Predicted dynamics = sin-like wave

Inaccurate Component (Ex. g_1^0, g_3^0):

Incorrect positive and negative sign

$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

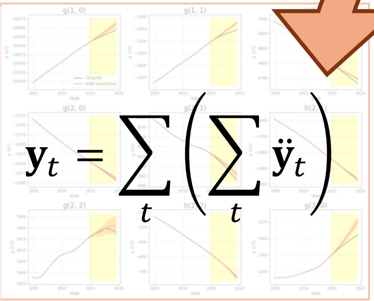


— **Training Data** (MCM model)

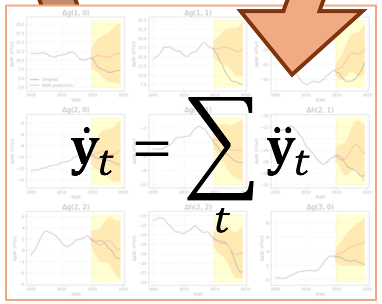
— **Validation Data** (MCM model)

— **RNN Prediction**

Summation



$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$



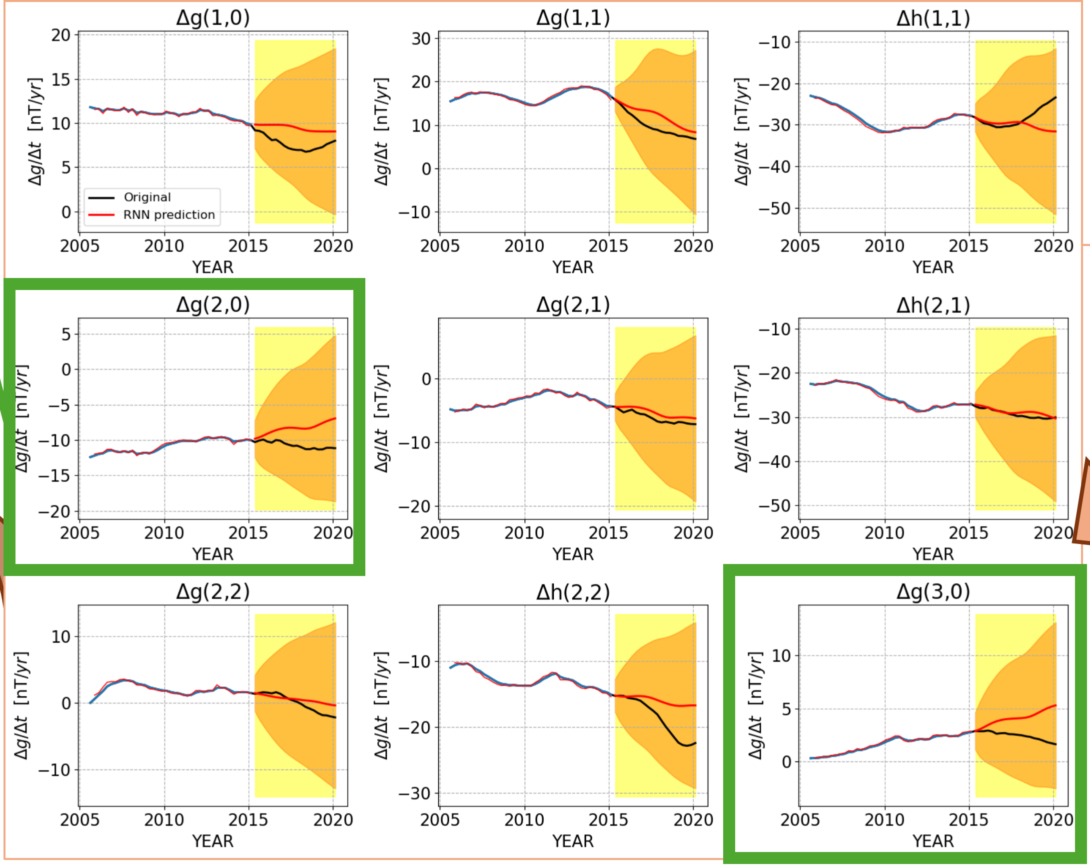
$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$

HINDCAST | 1^{st} order time derivative ($\dot{\mathbf{y}}$)

Common Feature:
~~Extends the last trend~~

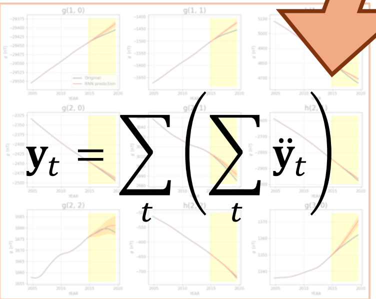
(Ex. g_2^0, g_3^0)
Inaccurate Component:
Inaccurate forecast caused by the sudden change of the trend in the validation period (2014).

$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$

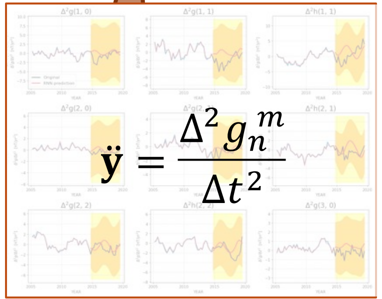


— **Training Data** (MCM model)
— **Validation Data** (MCM model)
— **RNN Prediction**

Summation



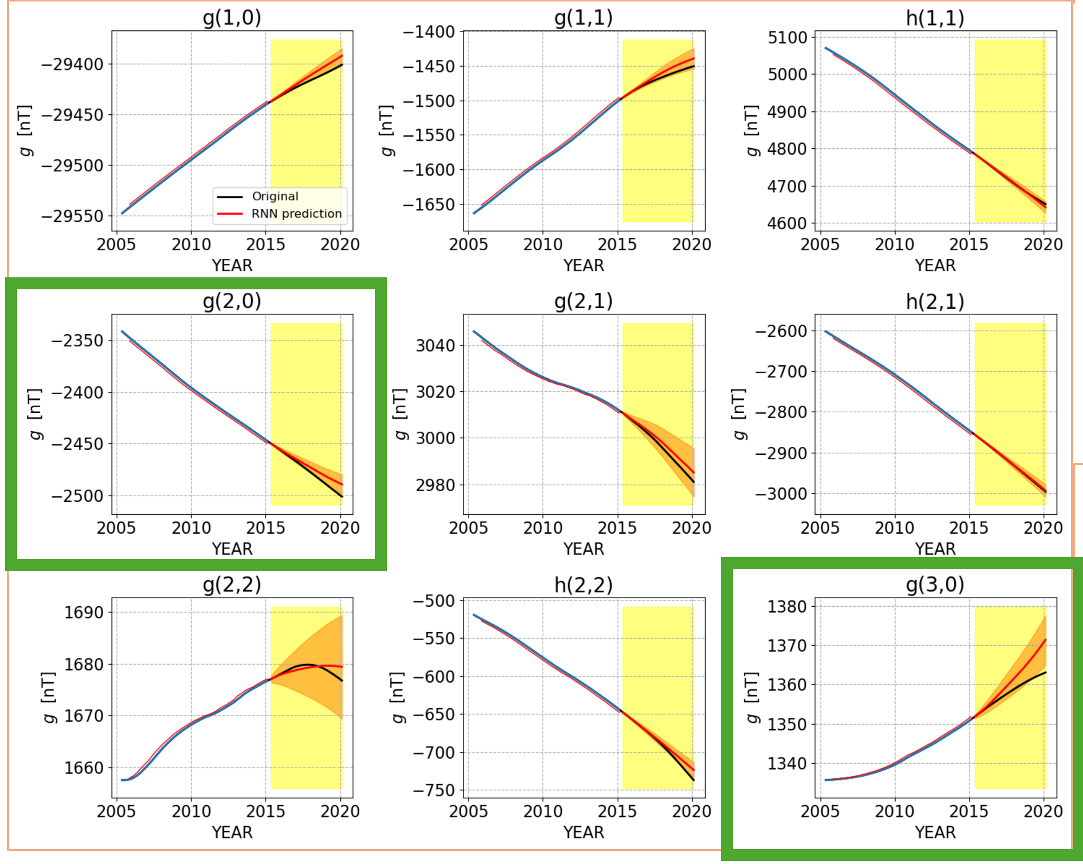
$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$



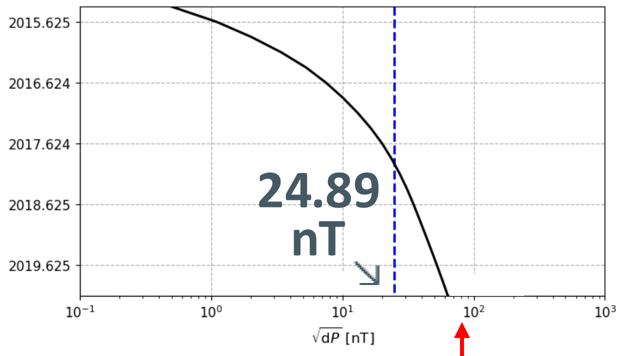
$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

HINDCAST | Original time series (y)

$$\mathbf{y}_t = \sum_t \left(\sum_t \ddot{\mathbf{y}}_t \right)$$

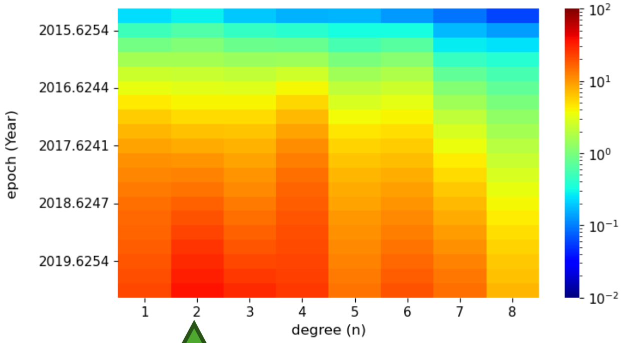


Total Error $\sqrt{dP} = \sqrt{\sum_n dP_n}$



62.78 nT

Degree dependence $\sqrt{dP_n} = \sqrt{\sum_m (n+1)[g^m_{\mathrm{TEST}} - g^m_{\mathrm{PRED}}]^2}$



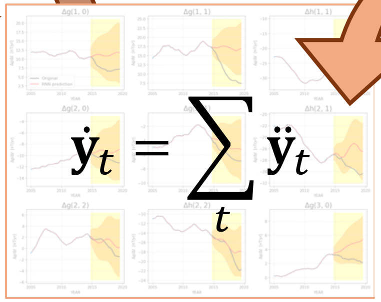
$n = 2$

Training Data (MCM model)

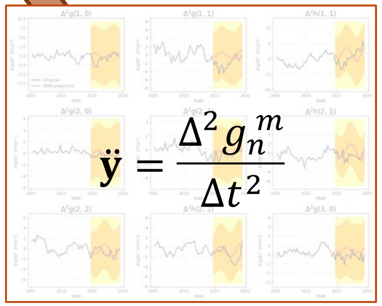
Validation Data (MCM model)

RNN Prediction

Summation



$$\dot{\mathbf{y}}_t = \sum_t \ddot{\mathbf{y}}_t$$



$$\ddot{\mathbf{y}} = \frac{\Delta^2 g_n^m}{\Delta t^2}$$

HINDCAST | RNN hidden layer

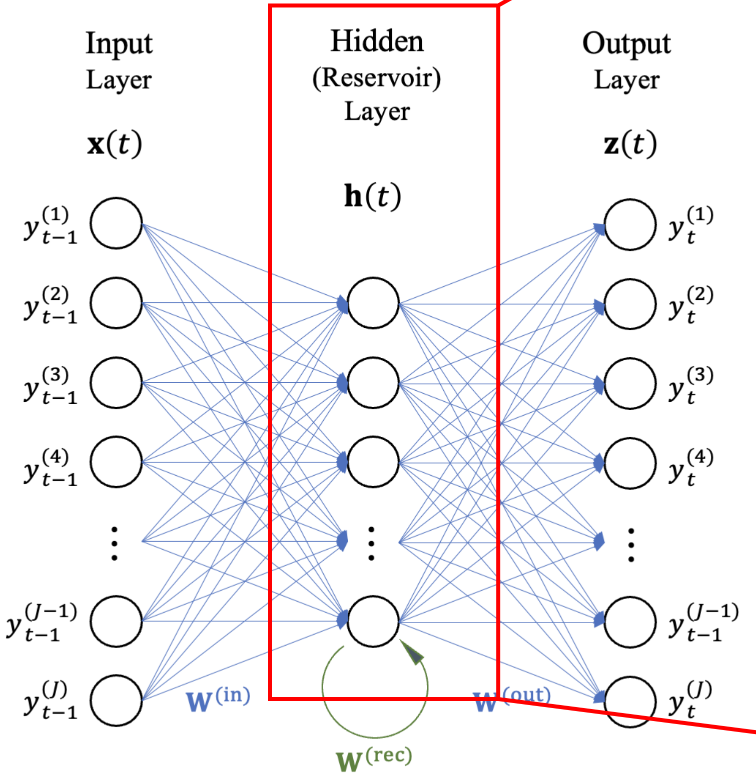
Elman Network

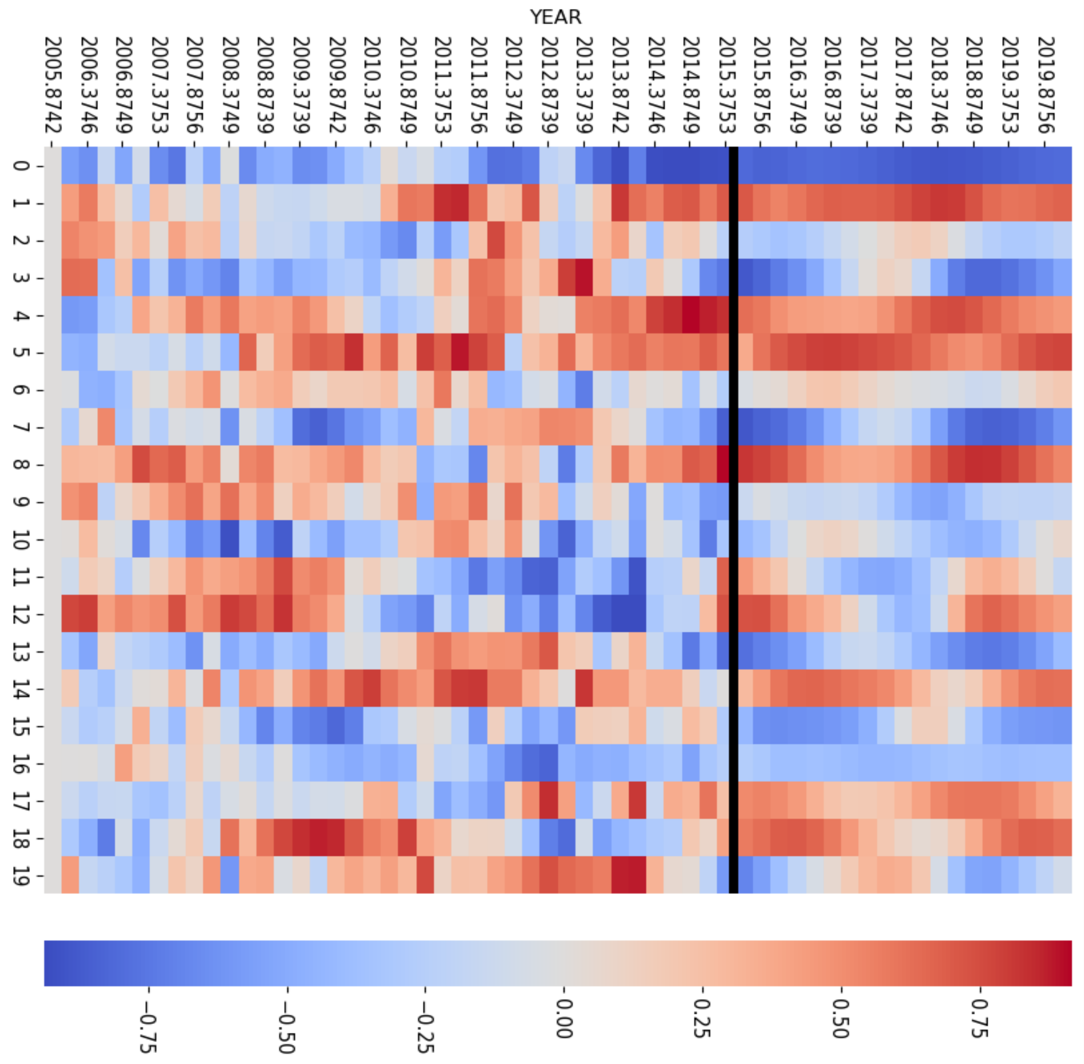
RNN cell

$$\mathbf{h}(t) = \tanh\left(\mathbf{W}^{(\mathrm{in})} \cdot \mathbf{x}(t) + \mathbf{W}^{(\mathrm{rec})} \cdot \mathbf{h}(t-1) + \mathbf{b}^{(\mathrm{rec})}\right)$$

Affine (Dense)

$$\mathbf{z}(t) = \mathbf{W}^{(\mathrm{out})} \cdot \mathbf{h}(t) + \mathbf{b}^{(\mathrm{out})}$$





Appendix | Gauss coefs / Forecast Error Chart

$$\begin{cases} \mathrm{rot}\vec{B} = \vec{0} \\ \mathrm{div}\vec{B} = 0 \end{cases} \Longleftrightarrow \begin{cases} \vec{B} = -\mathrm{grad}\ U \\ \mathrm{div\ grad}\ U = -\nabla^2 U = 0 \end{cases}$$

$$U(r,\theta,\phi) = R_E \sum_{n=1}^{\infty}\sum_{m=0}^{n}\left\{(g_n^m \cos m\phi + h_n^m \sin m\phi)\left(\frac{R_E}{r}\right)^{n+1}\right\}P_n^m(\cos\theta) + R_E \sum_{n=1}^{\infty}\sum_{m=0}^{n}\left\{(q_n^m \cos m\phi + s_n^m \sin m\phi)\left(\frac{r}{R_E}\right)^{n}\right\}P_n^m(\cos\theta)$$

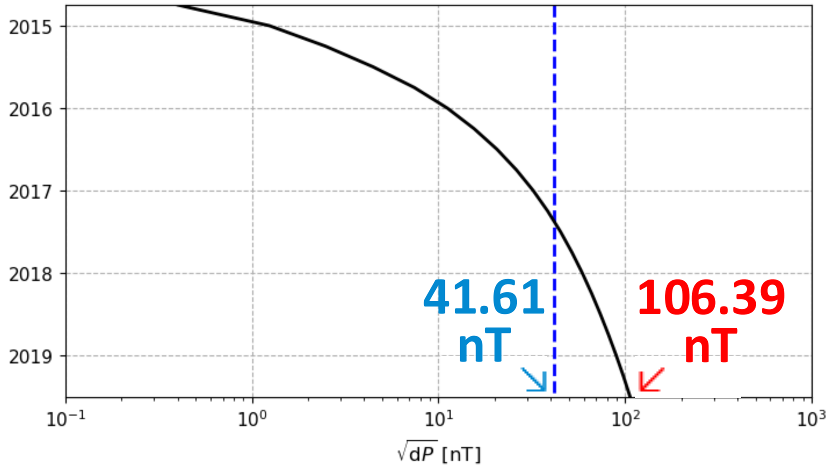
Degree dependence of Error : $\sqrt{dP_n} = \sqrt{\sum_m (n+1)[g^m_{\mathrm{TEST}} - g^m_{\mathrm{PRED}}]^2}$

Total Error : $\sqrt{dP} = \sqrt{\sum_n dP_n}$

Mauersberger-Lowes spectrum

(Langel and Estes, 1982)

Total Error



Degree dependence of Error