NeGIcourse_2019_Jakob_Boyd_Pernov

January 6, 2020

Comparison of Atmospheric Mercury Depletion Events at Villum Research Station and Zeppelin Mountain

Climate science at high latitudes: eScience for linking Arctic measurements and modeling

Jakob Boyd Pernov (jbp@evns.au.dk) Department of Environmental Science and iClimate at Aarhus University, Denmark Group 1: Marit Kollstuen and He Xucheng Abstract The Polar Regions experience depletion events of mercury every Spring, following Polar sunrise. These depletion events can introduce mercury into the ecosystem, thus posing a risk to aquatic as well as human health (Lehnherr, 2014). Understanding this behavior can help predict how mercury will respond to a changing climate in the Arctic. To address this, depletions events were investigated at two High Arctic sites: Villum Research Station and Zeppelin Mountain. The differences in sea ice and temperature can help elucidate the behavior of mercury and possible sources of reactive halogens.

Villum Research Station experiences more sea ice and lower temperatures, it has a lower annual mean of mercury, higher magnitude of reemission of mercury during the summer, and snowpack recycling with contributions from refreezing leads is proposed as a possible sources of reactive halogens. Zeppelin Mountain, surrounded by considerably less sea ice and has warmer temperatures, experiences a higher annual mean of mercury, less magnitude of reemission of mercury during the summer, and sea-salt aerosol is proposed as a possible source of reactive halogens. Zeppelin Mountain was also demonstrated to be influenced by intrusion of mercury-depleted air from aloft.

While much is still unknown about the dynamics of mercury oxidation in the Arctic, this work can help infer the response of mercury and its oxidation in a warming Arctic.

1 Introduction

Gaseous elemental mercury (GEM) is a ubiquitous pollutant in the atmosphere due to its long residence time. The sources of mercury include anthropogenic emissions (fossil fuel combustion and small artisan gold mines) and natural emissions (volcanoes, biomass burning, ocean and soil evasion, and reemission of deposited mercury) (AMAP, 2011). In locations with elevated halogen concentrations (Polar environments, coastal regions, volcanic plumes, and salt lakes) GEM can be quickly oxidized to its divalent form (HgÌI) or more commonly, Gaseous Oxidized Mercury (GOM). Once formed, GOM can either bind to aerosol particles, becoming particulate bound mercury (PHg), or deposit onto the snowpack. The fast oxidization and removal of GEM is referred to atmospheric mercury depletion events (AMDEs). AMDEs occur every Spring following Polar sunrise through a series of photochemically induced halogen explosions (Simpson et al., 2015) (R1-R4) followed by mercury oxidization (Lindberg et al., 2002).

X2 + h 2X (R1)

X + O3 XO + O2 (R2)

XO + HO2 HOX + O2 (R3)

HOX + X-(aq) + H+(aq) X2 +H20 (R4)

Where X = Cl, Br, or I and (aq) denotes the aqueous phase. The sources of these halogens are considered to be snowpack recycling, sea-salt aerosol, sea ice/refreezing leads, and photolysis of halogen reservoir species (Simpson et al., 2007; Simpson et al., 2015; Custard et al., 2017; Peterson et al., 2019). This halogen explosion mechanism requires an acidified heterogeneous surface (i.e., snowpack or aerosol surfaces), cold temperatures, and sunlight. Mercury oxidization is hypothesized to initiate with halogens Br through reactions (R6-R7) (Steffen et al., 2008).

Br + Hg0 HgBr (R5)

HgBr + Y HgBrY (R6)

Where Y could be OH, O3, NO2, NO, HO2, Br, Cl, BrO, ClO, I, IO. The exact chemical formula for GOM and PHg is currently unknown so they are operationally defined by their detection methods (Steffen et al., 2008).

Once deposited onto the Earth's surface, GEM, GOM, and PHg can be methylated through biotic and abiotic processes to organic mercury (methyl and dimethyl mercury). Organic mercury is an extremely powerful neurotoxin that bio-accumulates in upper trophic levels thus posing harmful effects to ecosystems and human health (especially indigenous peoples in high latitudes) (Park and Zheng, 2012). Therefore, it is pertinent to understand mercury oxidation in response to a changing climate, especially in high latitude regions.

This study investigates AMDEs at two High Arctic sites: Villum Research Station and Zeppelin Mountain station. These sites were investigated by comparison of meteorological parameters (short-wave down welling radiation (SWD), temperature, relative humidity (RH)) as well as sea ice area fraction (SI). The methods of the investigation will be described in section 2. The results will be presented and discussed in section 3, including occurrence of AMDEs at both sites, connections to meteorological data, and case studies of AMDEs occurring outside of Spring.

2 Methods

Two High Arctic sites were compared for this investigation, Villum Research Station (VRS, 81.6ř N 16.67ř W, 24 m above sea level) and Zeppelin Mountain Station (Zep, 79.93ř N 11.50ř E, 475 m above sea level) for the years 2011-2014. These two stations are within the Polar Dome year-round (Bozem et al., 2019) and are separated by ~700 km; however, they experience different meteorological conditions due to the North Atlantic current and the flow of ice out of the Arctic Ocean. This allows for a meaningful analysis of the behavior of mercury oxidation between these sites.

GEM was analyzed at both sites via pre-concentration on a gold trap followed by thermal desorption and detection by cold vapor atomic fluorescence spectroscopy (CVAFS) (Skov et al., 2004; Sprovieri et al., 2005). Mercury concentrations for VRS and Zep were retrieved from the Aarhus University's Database and the EMEP database (ebas.nilu.no, Tørseth et al., 2012), respectively. AMDEs were defined as GEM concentrations below 0.5 ng m-3 and three (two) consecutive decreasing measurements for VRS (Zep), due to different temporal resolutions (one hour for Zep and 30 minutes for VRS). Modelled data for VRS (SI and SWD) and Zep (SI, SWD, Temperature, and Relative Humidity) were obtained via the CMIP6 database (esgf-node.llnl.gov/search/cmip6/). SI was modelled via the NICAM 16-7S model (Tomita and Satoh, 2004) and SWD, temperature, and RH was modelled via the CESMS model (Gettelman et al., 2018). SI and SWD were modelled globally but were masked for ś4ř longitude and latitude around VRS and Zep.

Air mass back trajectory analysis was accomplished through use of the HYSPLIT Trajectory Model (Draxler et al., 1998) using Global Data Assimilation System (GDAS) meteorological data on a 1ř resolution. Trajectories were calculated at a height of 50 m above ground level and for 72 hours backwards in time.

3 Results & Discussion

3.1. Occurrence of AMDEs

AMDEs are commonly observed in the every year after Polar sunrise in the Spring (February through May with usually few events in June) (Steffen et al., 2015; Skov et al., 2004). This pattern is also observed for VRS and Zep (Fig. 1). Zep experienced more AMDEs during this period with 45 vs 38 for VRS. The occurrence of AMDEs for VRS (Fig 1., top panel) is highest for the month of April followed by May, March, and lastly February. For Zep (Fig 1., bottom panel), April and May both have the highest occurrence of AMDEs at 17 each, followed by March, June, and January. The presence of AMDEs in January and June are of interest and will be explored further in section 3.3.

Figure 1. Monthly Occurrence of AMDEs (number of AMDE days (x-axis) versus each month (y-axis)) at VRS (top panel) and Zeppelin Mountain (bottom panel) for the years 2011-2014.

3.2. Meteorological Parameters during AMDEs To gain a better understanding of the environmental conditions during individual AMDEs at each site, meteorological parameters (SWD, SI, temperature, and RH) a mean value for each AMDE day was calculated (Table A1 and A2). These values were then averaged for each month ad AMDE was observed in order to obtain a clearer picture of the difference between each site (Table 1 for Zep, Table 2 for VRS).

Table 1. Monthly mean for meteorological parameters at Zep for the analysis period (2011-2014). Month SWD (W m-2) Temperature (řC) RH (%) SI (%) January 0 -12.8 55 25 March 57 -17.5 65 26 April 116 -11.1 67 33 May 235 -7.1 64 37 June 255 0.3 79 29

Table 2. Monthly mean for meteorological parameters at VRS for the analysis period (2011-2014). Month SWD (W m-2) Temperature (řC) RH (%) SI (%) February 1 -26.7 70 99 March 43 -29.3 68 99 April 98 -25.0 67 99 May 207 -12.2 74 98

An assessment between Table 1 and Table 2 shows that Zep experiences more SWD, warmer temperatures, slightly lower RH values, and considerably lower amounts of SI. The differences in these meteorological parameters are largely driven by the North Atlantic Current bringing warm waters to Svalbard, while the East Greenlandic Current transports cold waters and sea ice south of the Arctic basin to the east coast of Greenland (Saloranta and Haugan 2001). The most drastic difference observed between VRS and Zep is SI. This will also drive part of the temperature difference, with sea ice preventing the exchange of heat from the relatively warm waters below with the cold air above. SI also controls the direct release of halogen species between the ocean-atmosphere interface.

To investigate the differences in SI, the daily mean value of GEM and SI for the analysis period (2011-2014) are visualized in Figures 2 and 3. Two key dissimilarities between the sites are evident from Figures 2 and 3. The yearly mean value is larger at Zep compared to VRS (1.50 vs 1.45 ng m-3) and the magnitude of reemission in the summer is lower for Zep compared to VRS (mean (max) GEM concentration for June – August for Zep and VRS is 1.59 (2.94) vs 1.69 (3.48) ng m-3, respectively). These dissimilarities are hypothesized to be primarily governed by the large differences in SI. The lower fraction of SI and higher temperatures at Zep is conducive for GEM evasion from the ocean, as the northern Atlantic has been demonstrated to be a source of GEM (Sørensen et al., 2010) although Svalbard is closer to mainland Europe and could be affected by transport of GEM to the Arctic (Hirdman et al., 2009). These are likely confounding factors in the higher GEM concentrations found at Zep. The presence of SI at VRS allows for the retention of deposited mercury in the snowpack overlaying the sea ice while the open waters surrounding Zep would remove any deposited mercury to the ocean. Thus, the lower magnitude of mercury reemission in the summer can be expounded by the absence of large areas of SI surrounding Zep.

Figure 2. Daily average of Hg (GEM) and SI for the years 2011-2014 at Zeppelin Mountain. Error bars are represented by the standard deviation of the daily mean value.

Figure 3. Daily average of Hg (GEM) and SI for the years 2011-2014 at Villum Research Station.

Error bars are represented by the standard deviation of the daily mean value.

The seasonal pattern of SI and the behavior of mercury between the two sites offers insight into the sources of the reactive halogens responsible for these AMDEs. VRS is mainly ice-locked throughout the year thereby providing more spatial area for snowpack accumulation and high probability of forming open leads. VRS also experiences colder temperatures, which suppresses snow melting as well as sea-salt aerosol production. Theses colder temperatures and large amounts of SI also create conditions for refreezing leads to contribute to the atmospheric burden of reactive halogens. These two factors create an environment favorable for a combination of snowpack recycling and refreezing leads to be dominant sources of reactive halogens (Peterson et al., 2018; Custard et al., 2017; Peterson et al., 2019). In contrast to VRS, Zep experiences more open water, warmer temperatures, and more SWD. This creates conditions that promotes sea-salt aerosol production (Vogt, Crutzen and Sander, 1996). The source of halogen radicals at VRS is proposed to be snowpack recycling of halogens (Peterson et al., 2018). The source of reactive halogens at Zep is proposed to be sea-salt aerosol emitted from the open ocean (Simpson et al., 2015).

3.3. AMDEs occurring outside of Spring As stated in section 3.1, AMDEs are typically observed from February to May with few events in June (Steffen et al., 2008). In light of this, the events observed in January and June at Zep were further probed. The January event is suspicious as it is during Polar night with no sunlight available for halogen activation. An analysis of GEM concentrations (Fig A1) during those days (January 7-8, 2014) yielded trends similarly observed during AMDEs during Spring (constant depletion and not several anomalous values below 0.5 ng m-3). The air mass history for this day was examined through HYSPLIT back trajectories (Fig 4). For the entire depletion event air mass consistently arrived from the north, over the Arctic Ocean, and from aloft. The vertical profiles of the back trajectories indicate upper troposphere and possibly lower stratosphere intrusion.

Figure 4. Back trajectory analysis for January 7 and 8, 2014 for air masses arriving at 50 meters above ground level for Zeppelin Mountain station. A new trajectory was calculated every 12 hours for the duration of the AMDE indicated by the different colors. The height on each trajectory above ground level is plotted in the lower portion.

For the AMDEs observed in June (June 7,9,10, 2013 and June 14, 2014), HYSPLIT back trajectories are displayed in Figure 5, left panel for June 2013 events and right panel for the June 2014 event. For the June 7, 2013 event, air masses consistently arrived from the southwest after traversing Greenland and had substantial surface contact. For the June 9 and 10, 2013 events air masses consistently arrived from the northwest but also had substantial surface contact. The June 14, 2014 event did not experience as consistent air mass history although all air masses originated from the northwest and had considerably surface contact.

Figure 5. Back trajectories for June 11, 2013 (left panel) and June 15, 2014 (right panel) started at 50 m above ground level. For the June 2013 event, a new trajectory was calculated every 12 hours for the previous three days. For the June 2014 event, a new trajectory was calculated every 6 hours indicated by the different colors. The height on each trajectory above ground level is plotted in the lower portion.

The air mass history of AMDEs observed in June gives credence to the hypothesis of sea-salt aerosol as the source of reactive halogens. The extensive surface contact permits the entrainment of sea-salt aerosols emitted from the open ocean. June is after the onset of snowpack melt, Burd et al. (2017) demonstrated the onset of the melt season hampers the recycling of BrO on snowpack surfaces. While other sources of reactive halogens in the marine boundary layer exist (see Introduction), the data presented here indicates sea-salt aerosol as a major source at Zep.

4 Conclusions GEM is a global pollutant that, once oxidized and deposited, can be transformed into a powerful neurotoxin, harmful to both ecosystems and human health. Understanding its ox-

idation behavior is therefore an important aspect of atmospheric chemistry in the High Arctic. This analysis compares AMDEs between VRS and Zep in relation to meteorological parameters. It is shown that during AMDEs, VRS and Zep differ in terms of temperature and SI, with Zep having warmer temperatures and considerably less SI than VRS. These differences can explain the different behavior of GEM and provide insight into the possible sources of reactive halogens at these two sites. Zep, having more open waters, experiences a higher annual mean of GEM, possibly due to ocean evasion or GEM transport, and a lower magnitude of reemission in the summer. These warmer temperatures and open waters promote sea-salt aerosol production. It is therefore proposed that the source of reactive halogens at Zep is sea-salt aerosol. Zeppelin Mountain was also demonstrated to be influenced by intrusion of mercury-depleted air from aloft. VRS, being ice-locked throughout the year, has a lower annual mean of GEM and a higher magnitude of reemission due to increased snowpack surface on sea ice. This provides a surface for GOM deposition and reemission as well as acting as a heterogeneous surface for halogen recycling. The sources of reactive halogens at VRS is hypothesized to be snowpack recycling with contributions from refreezing leads. The Arctic is expected to increase in temperature in the coming years, this will be accompanied by less sea ice and a longer melt season. These changes will affect the oxidation of GEM and thus the frequency of AMDEs. The hypotheses of this work can offer understanding of how these two sites will respond to a warming Arctic. Future work should expand on this analysis by incorporating more meteorological parameters and a longer time series.

5 References AMAP Assessment 2011: Mercury in the Arctic. Arctic Monitoring and Assessment Programme (AMAP), Oslo, Norway. xiv + 193 pp. Bozem, H., Hoor, P., Kunkel, D., Köllner, F., Schneider, J., Herber, A., Schulz, H., Leaitch, W., Aliabadi, A., Willis, M., Burkart, J. and Abbatt, J. (2019). Characterization of Transport Regimes and the Polar Dome during Arctic Spring and Summer using in-situ Aircraft Measurements. Atmospheric Chemistry and Physics Discussions, pp.1-36. Burd, J., Peterson, P., Nghiem, S., Perovich, D. and Simpson, W. (2017). Snowmelt onset hinders bromine monoxide heterogeneous recycling in the Arctic. Journal of Geophysical Research: Atmospheres, 122(15), pp.8297-8309. Cover Photo of Zeppelin: http://agage.mit.edu/stations/zeppelin-ny-ålesund Custard, K. D., Raso, A. R. W., Shepson, P., Staebler, R., & Pratt, K. A. (2017). Production and release ofmolecular bromine and chlorine from the Arctic coastal snowpack. ACS Earth & Space Chemistry, 1(3), 142–151. Draxler, R., Roland & Hess, G. (1998). An overview of the HYSPLIT_4 modeling system for trajectories, dispersion, and deposition. Australian Meteorological Magazine. 47. 295-308. Gettelman, A., Callaghan, P., Larson, V., Zarzycki, C., Bacmeister, J., Lauritzen, P., Bogenschutz, P. and Neale, R. (2018). Regional Climate Simulations With the Community Earth System Model. Journal of Advances in Modeling Earth Systems, 10(6), pp.1245-1265. Hirdman, D., Aspmo, K., Burkhart, J., Eckhardt, S., Sodemann, H. and Stohl, A. (2009). Transport of mercury in the Arctic atmosphere: Evidence for a spring-time net sink and summer-time source. Geophysical Research Letters, 36(12). Lehnherr, I. (2014). Methylmercury biogeochemistry: a review with special reference to Arctic aquatic ecosystems. Environmental Reviews, 22(3), pp.229-243. Lindberg, S., Brooks, S., Lin, C., Scott, K., Landis, M., Stevens, R., Goodsite, M. and Richter, A. (2002). Dynamic Oxidation of Gaseous Mercury in the Arctic Troposphere at Polar Sunrise. Environmental Science & Technology, 36(6), pp.1245-1256. McNamara, S., W. Raso, A., Wang, S., Thanekar, S., Boone, E., Kolesar, K., Peterson, P., Simpson, W., Fuentes, J., Shepson, P. and Pratt, K. (2019). Springtime Nitrogen Oxide-Influenced Chlorine Chemistry in the Coastal Arctic. Environmental Science & Technology, 53(14), pp.8057-8067. Michalowski, B. A., Francisco, J. S., Li, S.-M., Barrie, L. A., Bottenheim, J. W., & Shepson, P. B. (2000). A computer model study of multiphase chemistry in the Arctic boundary layer during polar sunrise. Journal of Geophysical Research, 105(D12), 15,131–15,145. Park, J. and Zheng, W. (2012). Human Exposure and Health Effects of Inorganic and Elemental Mercury. Journal

of Preventive Medicine & Public Health, 45(6), pp.344-352. Peterson, P., Hartwig, M., May, N., Schwartz, E., Rigor, I., Ermold, W., Steele, M., Morison, J., Nghiem, S. and Pratt, K. (2019). Snowpack measurements suggest role for multi-year sea ice regions in Arctic atmospheric bromine and chlorine chemistry. Elem Sci Anth, 7(1), p.14. Saloranta, T. and Haugan, P. (2001). Interannual variability in the hydrography of Atlantic water northwest of Svalbard. Journal of Geophysical Research: Oceans, 106(C7), pp.13931-13943. Simpson, W., Brown, S., Saiz-Lopez, A., Thornton, J. and von Glasow, R. (2015). Tropospheric Halogen Chemistry: Sources, Cycling, and Impacts. Chemical Reviews, 115(10), pp.4035-4062. Simpson, W., von Glasow, R., Riedel, K., Anderson, P., Ariya, P., Bottenheim, J., Burrows, J., Carpenter, L., FrieSS, U., Goodsite, M., Heard, D., Hutterli, M., Jacobi, H., Kaleschke, L., Neff, B., Plane, J., Platt, U., Richter, A., Roscoe, H., Sander, R., Shepson, P., Sodeau, J., Steffen, A., Wagner, T. and Wolff, E. (2007). Halogens and their role in polar boundary-layer ozone depletion. Atmospheric Chemistry and Physics, 7(16), pp.4375-4418. Skov, H., Christensen, J., Goodsite, M., Heidam, N., Jensen, B., Wåhlin, P. and Geernaert, G. (2004). Fate of Elemental Mercury in the Arctic during Atmospheric Mercury Depletion Episodes and the Load of Atmospheric Mercury to the Arctic. Environmental Science & Technology, 38(8), pp.2373-2382. Sørensen, A., Skov, H., Jacob, D., Sørensen, B. and Johnson, M. (2010). Global Concentrations of Gaseous Elemental Mercury and Reactive Gaseous Mercury in the Marine Boundary Layer. Environmental Science & Technology, 44(19), pp.7425-7430. Steffen, A., Douglas, T., Amyot, M., Ariya, P., Aspmo, K., Berg, T., Bottenheim, J., Brooks, S., Cobbett, F., Dastoor, A., Dommergue, A., Ebinghaus, R., Ferrari, C., Gardfeldt, K., Goodsite, M., Lean, D., Poulain, A., Scherz, C., Skov, H., Sommar, J. and Temme, C. (2008). A synthesis of atmospheric mercury depletion event chemistry in the atmosphere and snow. Atmospheric Chemistry and Physics, 8(6), pp.1445-1482. Steffen, A., Lehnherr, I., Cole, A., Ariya, P., Dastoor, A., Durnford, D., Kirk, J. and Pilote, M. (2015). Atmospheric mercury in the Canadian Arctic. Part I: A review of recent field measurements. Science of The Total Environment, 509-510, pp.3-15. Tomita, H. and Satoh, M. (2004). A new dynamical framework of nonhydrostatic global model using the icosahedral grid. Fluid Dynamics Research, 34(6), pp.357-400. Tørseth, K., Aas, W., Breivik, K., Fjæraa, A., Fiebig, M., Hjellbrekke, A., Lund Myhre, C., Solberg, S. and Yttri, K. (2012). Introduction to the European Monitoring and Evaluation Programme (EMEP) and observed atmospheric composition change during 1972-2009. Atmospheric Chemistry and Physics, 12(12), pp.5447-5481. Vogt, R., Crutzen, P. and Sander, R. (1996). A mechanism for halogen release from sea-salt aerosol in the remote marine boundary layer. Nature, 383(6598), pp.327-330.

6 Appendix

Table A1. Daily mean for meteorological parameters during AMDEs at Zep. Date SWD (W m-2) Temperature (řC) RH (%) SI (%) 24-03-2011 41 -18.3 65 37 25-03-2011 57 -17.5 68 37 09-05-2011 214 -6.9 79 33 10-05-2011 300 -7.9 64 33 20-05-2011 239 -5.2 80 32 21-05-2011 334 -4.1 68 32 23-05-2011 201 -1.9 85 32 01-04-2012 60 -14.8 55 27 05-04-2012 90 -15.6 67 30 02-05-2012 119 -8.7 80 32 24-03-2013 59 -16.4 54 26 25-03-2013 62 -17.8 56 26 06-04-2013 95 -13.6 55 30 07-04-2013 122 -16.7 51 31 09-04-2013 136 -17.1 52 31 14-04-2013 116 -11.1 75 34 25-04-2013 85 -2.7 93 33 26-04-2013 75 -6.4 90 33 27-04-2013 122 -8.1 82 33 28-04-2013 160 -7.6 73 33 25-05-2013 298 -4.0 60 33 26-05-2013 222 -3.6 69 33 07-06-2013 347 0.7 73 28 09-06-2013 233 1.3 79 29 10-06-2013 262 -0.1 79 29 07-01-2014 0 -12.3 54 25 08-01-2014 0 -13.3 56 25 24-03-2014 33 -13.0 79 23 04-04-2014 63 -12.7 62 30 05-04-2014 77 -14.1 64 31 17-04-2014 131 -9.5 68 36 18-04-2014 107 -13.4 72 36 27-04-2014 159 -9.4 75 36 28-04-2014 233 -9.1 61 36 30-04-2014 178 -7.8 64 36 01-05-2014 184 -7.1 61 37 02-05-2014 235 -7.4 60 37 10-05-2014 262 -6.5 59 37 11-05-2014 195 -9.2 63 37 12-05-2014 265 -8.9 64 37 13-05-2014 273 -9.1 61 37 14-05-2014 211 -8.9 59 37 15-05-2014 285 -7.7 65 37 21-05-2014 138 -3.8 83 37 14-06-2014 248 -0.7 79 36

Table A2. Daily mean for meteorological parameters during AMDEs at VRS. Date SWD (W

m-2) Temperature (řC) RH (%) SI (%) 23-03-2011 38 -18.9 75 99 24-03-2011 45 -20.1 71 99 05-04-2011 67 -25.0 67 99 17-04-2011 156 -25.0 67 99 14-05-2011 270 -12.2 74 98 15-05-2011 314 -14.3 69 98 16-05-2011 318 -13.1 55 98 19-05-2011 198 -10.5 79 98 20-05-2011 203 -9.8 83 98 23-03-2012 44 -28.7 69 99 26-03-2012 50 -34.0 63 99 27-03-2012 43 -30.0 68 99 29-03-2012 53 -29.3 68 99 02-04-2012 60 -23.8 74 99 03-04-2012 69 -30.8 66 99 07-04-2013 97 -28.1 66 99 16-04-2013 128 -26.1 65 99 17-04-2013 98 -25.3 64 99 18-04-2013 151 -25.7 64 99 27-04-2013 158 -19.5 58 99 08-05-2013 191 -16.5 69 99 09-05-2013 199 -17.5 74 99 24-05-2013 248 -8.0 81 97 25-05-2013 264 -7.0 70 97 26-02-2014 1 -26.7 70 99 01-03-2014 2 -32.1 63 99 16-03-2014 23 -33.4 61 100 26-03-2014 31 -24.8 72 100 05-04-2014 88 -21.7 72 100 06-04-2014 88 -25.5 69 100 09-04-2014 73 -24.5 67 100 10-04-2014 70 -27.8 64 100 16-04-2014 113 -24.4 69 100 17-04-2014 127 -24.3 70 100 18-04-2014 121 -21.2 67 100 25-04-2014 141 -20.8 69 99 01-05-2014 205 -15.0 80 99 13-05-2014 207 -11.7 73 99

Figure A1. GEM concentrations for January 7 & 8, 2014 at Zep.

[ ]:
```
## Met Data for Zeppelin Mountain Station
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline

pathname='shared-ns1000k/dataporten-home/
 ↪b8b982ef-2de983-2d4813-2dbbd2-2d5e4e04927306/'
match_date_file = pathname + 'AMDE_NYA.txt'
date_data = pd.read_csv(match_date_file, header = None,
                    parse_dates = True)
date_data.columns = ['Date/Time']
date_data['Date/Time'] = pd.to_datetime(date_data['Date/Time'])
date_data = date_data.set_index('Date/Time')

def find_start(file):
    i = 1
    with open(file, 'r') as input:
        for line in input:
            if '*/' in line:
                break
            i = i + 1
    return i

def file_reader(filename):
    skip_rows = find_start(filename)
    data = pd.read_csv(filename, sep='\t', skiprows=skip_rows,
                    index_col = 'Date/Time',
                    dtype={'DIR [W/m**2]': 'float64',
                           'LWD [W/m**2]': 'float64'},
                    parse_dates = True)
    daily_data=data.resample('D').mean()
    daily_data = daily_data.loc[date_data.index].dropna()
    return daily_data
```

```
path = 'shared-ns1000k/inputs/BSRN_NYA/'
filename = filename = path + 'NYA_radiation_'+ str(date_data.index[0].year) +␣
 ↪'-0' + str(date_data.index[0].month) + '.tab'
daily_data = file_reader(filename)
for i in range(1,date_data.index.shape[0]):
    filename = path + 'NYA_radiation_'+ str(date_data.index[i].year) + '-0' +␣
 ↪str(date_data.index[i].month) + '.tab'
    daily_data = daily_data.append(file_reader(filename))

daily_data=daily_data.drop_duplicates(keep='first')
```

[ ]:
```
# Met Data for VRS
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline

pathname='shared-ns1000k/dataporten-home/
 ↪b8b982ef-2de983-2d4813-2dbbd2-2d5e4e04927306/'
match_date_file = './AMDE_VRS.txt'
date_data = pd.read_csv(match_date_file, header = None)
date_data.columns = ['Date/Time']
date_data['Date/Time'] = pd.to_datetime(date_data['Date/Time'])
date_data = date_data.set_index('Date/Time')

filename = 'VRSMet_11_14.txt'
data = pd.read_csv(filename, sep=',',
                   index_col = 'DateTime',
                   parse_dates = True)

daily_data=data.resample('D').mean()
AMDE_Met_VRS=daily_data.loc[date_data.index.tolist()].dropna()
```

[ ]:
```
#Met Data Zeppelin
import pandas as pd
%matplotlib inline

path = 'shared-ns1000k/inputs/BSRN_NYA/'
filename = path + 'NYA_radiation_2011-03.tab'

print(filename)

def find_start(file):
    i = 1
    with open(file, 'r') as input:
        for line in input:
```

```
            if '*/' in line:
                break
            i = i + 1
    return i

skip_rows = find_start(filename)
print(skip_rows)

data = pd.read_csv(filename, sep='\t', skiprows=skip_rows,
                   index_col = 'Date/Time',
                   dtype={'DIR [W/m**2]': 'float64',
                          'LWD [W/m**2]': 'float64'},
                   parse_dates = True)

data.head()
daily_data = data.resample('D').mean()
daily_data.to_csv("daily_NYA_radiation_2011-08.csv", index_label = ['Date/
 ↪Time'])

match_date_file = 'shared-ns1000k/dataporten-home/
 ↪b8b982ef-2de983-2d4813-2dbbd2-2d5e4e04927306/AMDE_VRS.txt'
date_data = pd.read_csv(match_date_file, header = None,
                   parse_dates = True)
date_data.columns = ['Date/Time']
date_data['Date/Time'] = pd.to_datetime(date_data['Date/Time'])
date_data = date_data.set_index('Date/Time')

for i in range(date_data.index.shape[0]):
    print(i,'NYA_radiation_' + str(date_data.index[i].year) + '-0' +␣
 ↪str(date_data.index[i].month) + '.tab')

daily_data.loc[date_data.index.tolist()].dropna()
```

[ ]:
```
# Met Data VRS
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
%matplotlib inline

pathname='shared-ns1000k/dataporten-home/
 ↪b8b982ef-2de983-2d4813-2dbbd2-2d5e4e04927306/'
match_date_file = './AMDE_VRS.txt'
date_data = pd.read_csv(match_date_file, header = None)
date_data.columns = ['Date/Time']
date_data['Date/Time'] = pd.to_datetime(date_data['Date/Time'])
date_data = date_data.set_index('Date/Time')
```

```
filename = 'VRSMet_11_14.txt'


data = pd.read_csv(filename, sep=',',
                   index_col = 'DateTime',
                   parse_dates = True)

daily_data=data.resample('D').mean()

daily_data=data.resample('D').mean()
AMDE_Met_VRS=daily_data.loc[date_data.index.tolist()].dropna()

AMDE_Met_VRS
```

[ ]:
```
# Rad data for VRS
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import cftime
import pandas as pd

#!wget http://esgf-data.ucar.edu/thredds/fileServer/esg_dataroot/CMIP6/CMIP/
 ↪NCAR/CESM2/amip/r1i1p1f1/day/rsds/gn/v20190218/
 ↪rsds_day_CESM2_amip_r1i1p1f1_gn_20100101-20150101.nc

path = 'rsds_day_CESM2_amip_r1i1p1f1_gn_20100101-20150101.nc'
SWD = xr.open_dataset(path)
SWD_wrking= SWD.sel(time=slice(cftime.DatetimeNoLeap(2011,1,1),cftime.
 ↪DatetimeNoLeap(2014,12,31)))
#SWD_wrking

pathname='shared-ns1000k/dataporten-home/
 ↪b8b982ef-2de983-2d4813-2dbbd2-2d5e4e04927306/'
match_date_file = './AMDE_VRS.txt'
date_data = pd.read_csv(match_date_file, header = None,parse_dates=True)
date_data.columns = ['Date/Time']
date_data['Date/Time'] = pd.to_datetime(date_data['Date/Time'])
date_data = date_data.set_index('Date/Time')

import sys
sys.path.append('negi-stuff/')
from negi_stuff.modules import make_folders

def get_wghts(lat):
    """
    get latitude weights for gaussian grid.
```

```
    :param lat: latitudes
    :return: weights
    """
    latr = np.deg2rad(lat) # convert to radians
    weights = np.cos(latr) # calc weights
    return weights


def masked_average(xa:xr.DataArray,
                   dim=None,
                   weights:xr.DataArray=None,
                   mask:xr.DataArray=None):
    """
    This function will average
    :param xa: dataArray
    :param dim: dimension or list of dimensions. e.g. 'lat' or␣
 ↪['lat','lon','time']
    :param weights: weights (as xarray)
    :param mask: mask (as xarray), True where values to be masked.
    :return: masked average xarray
    """
    #lest make a copy of the xa
    xa_copy:xr.DataArray = xa.copy()

    if mask is not None:
        xa_weighted_average = __weighted_average_with_mask(
            dim, mask, weights, xa, xa_copy
        )
    elif weights is not None:
        xa_weighted_average = __weighted_average(
            dim, weights, xa, xa_copy
        )
    else:
        xa_weighted_average =  xa.mean(dim)

    return xa_weighted_average



    # %% [markdown]


def __weighted_average(dim, weights, xa, xa_copy):
    '''helper function for masked_average'''
    _, weights_all_dims = xr.broadcast(xa, weights)  # broadcast to all dims
    x_times_w = xa_copy * weights_all_dims
    xw_sum = x_times_w.sum(dim)
```

```
    x_tot = weights_all_dims.where(xa_copy.notnull()).sum(dim=dim)
    xa_weighted_average = xw_sum / x_tot
    return xa_weighted_average


def __weighted_average_with_mask(dim, mask, weights, xa, xa_copy):
    '''helper function for masked_average'''
    _, mask_all_dims = xr.broadcast(xa, mask)  # broadcast to all dims
    xa_copy = xa_copy.where(np.logical_not(mask))
    if weights is not None:
        _, weights_all_dims = xr.broadcast(xa, weights)  # broadcast to all␣
 ↪dims
        weights_all_dims = weights_all_dims.where(~mask_all_dims)
        x_times_w = xa_copy * weights_all_dims
        xw_sum = x_times_w.sum(dim=dim)
        x_tot = weights_all_dims.where(xa_copy.notnull()).sum(dim=dim)
        xa_weighted_average = xw_sum / x_tot
    else:
        xa_weighted_average = xa_copy.mean(dim)
    return xa_weighted_average

def get_wghts(lat):
    """
    get latitude weights for gaussian grid.
    :param lat: latitudes
    :return: weights
    """
    latr = np.deg2rad(lat) # convert to radians
    weights = np.cos(latr) # calc weights
    return weights


# ## Application 1: Weigted global average:
# Grid cells have different area, so when we do the global average, they have␣
 ↪to be weigted by the area of each grid cell.
# Here we do it for 2 m temperature:

SWD_wrking_slice = SWD_wrking.sel(lat=slice(77, 85), lon=slice(340,348))

glob_mean_rad = masked_average(SWD_wrking_slice['rsds'], dim=['lat','lon'],␣
 ↪weights=get_wghts(SWD_wrking_slice['lat']))

glob_mean_rad_xr=xr.Dataset({'Rad': (('time'), glob_mean_rad)},
                            {'time':  glob_mean_rad.time}
                           )

glob_mean_rad_xr.to_netcdf('Sol_Rad_vrs_11_14.nc')
```

```
import xarray as xr
dset = xr.open_dataset('./Sol_Rad_vrs_11_14.nc')

aaa = pd.DataFrame({'time': glob_mean_rad_xr.time, 'rad': glob_mean_rad_xr.Rad})
aaa.to_csv('rad_vrs_all.csv')

list_times = []
for t in date_data.index:
    list_times.append(cftime.DatetimeNoLeap(t.year, t.month, t.day, 0, 0, 0, 0,␣
 ↪0, 0))

AMDE_RAD_VRS=dset.sel(time=list_times,method='nearest')

#AMDE_RAD_VRS.to_netcdf("AMDE_RAD_VRS.csv")
aaa = pd.DataFrame({'time': AMDE_RAD_VRS.time, 'rad': AMDE_RAD_VRS.Rad})

aaa.to_csv('AMDE_RAD_VRS.csv')
```

[ ]:
```
# Sea Ice Data for both sites
import xarray as xr
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import cftime
import os

# So this code is so ghetto. Anna wrote some of it, Sarah wrote some of it, and␣
 ↪I tried to make sense of it all.
# The sea ice data is in different years so I hard coded the files and ran them␣
 ↪individually for VRS and NYA.
# I know there is definitely better way to do this but again I am not a python␣
 ↪person.

path =␣
 ↪'siconc_SIday_NICAM16-7S_highresSST-present_r1i1p1f1_gr_20110101-20111231.nc'
sea_ice11 = xr.open_dataset(path)
sea_ice11['siconc'] = sea_ice11['siconca']
sea_ice11.drop('siconca')
sea_ice11=sea_ice11.drop('siconca')

path1 =␣
 ↪'siconc_SIday_NICAM16-7S_highresSST-present_r1i1p1f1_gr_20120101-20121231.nc'
sea_ice12 = xr.open_dataset(path1)

path2 =␣
 ↪'siconc_SIday_NICAM16-7S_highresSST-present_r1i1p1f1_gr_20130101-20131231.nc'
```

```
sea_ice13 = xr.open_dataset(path2)

path3 =␣
 ↪'siconc_SIday_NICAM16-7S_highresSST-present_r1i1p1f1_gr_20140101-20141231.nc'
sea_ice14 = xr.open_dataset(path3)

import sys
sys.path.append('negi-stuff/')
from negi_stuff.modules import make_folders

def get_wghts(lat):
    """
    get latitude weights for gaussian grid.
    :param lat: latitudes
    :return: weights
    """
    latr = np.deg2rad(lat) # convert to radians
    weights = np.cos(latr) # calc weights
    return weights


def masked_average(xa:xr.DataArray,
                   dim=None,
                   weights:xr.DataArray=None,
                   mask:xr.DataArray=None):
    """
    This function will average
    :param xa: dataArray
    :param dim: dimension or list of dimensions. e.g. 'lat' or␣
 ↪['lat','lon','time']
    :param weights: weights (as xarray)
    :param mask: mask (as xarray), True where values to be masked.
    :return: masked average xarray
    """
    #lest make a copy of the xa
    xa_copy:xr.DataArray = xa.copy()

    if mask is not None:
        xa_weighted_average = __weighted_average_with_mask(
            dim, mask, weights, xa, xa_copy
        )
    elif weights is not None:
        xa_weighted_average = __weighted_average(
            dim, weights, xa, xa_copy
        )
    else:
        xa_weighted_average =  xa.mean(dim)
```

```
    return xa_weighted_average



    # %% [markdown]


def __weighted_average(dim, weights, xa, xa_copy):
    '''helper function for masked_average'''
    _, weights_all_dims = xr.broadcast(xa, weights)  # broadcast to all dims
    x_times_w = xa_copy * weights_all_dims
    xw_sum = x_times_w.sum(dim)
    x_tot = weights_all_dims.where(xa_copy.notnull()).sum(dim=dim)
    xa_weighted_average = xw_sum / x_tot
    return xa_weighted_average


def __weighted_average_with_mask(dim, mask, weights, xa, xa_copy):
    '''helper function for masked_average'''
    _, mask_all_dims = xr.broadcast(xa, mask)  # broadcast to all dims
    xa_copy = xa_copy.where(np.logical_not(mask))
    if weights is not None:
        _, weights_all_dims = xr.broadcast(xa, weights)  # broadcast to all␣
 ↪dims
        weights_all_dims = weights_all_dims.where(~mask_all_dims)
        x_times_w = xa_copy * weights_all_dims
        xw_sum = x_times_w.sum(dim=dim)
        x_tot = weights_all_dims.where(xa_copy.notnull()).sum(dim=dim)
        xa_weighted_average = xw_sum / x_tot
    else:
        xa_weighted_average = xa_copy.mean(dim)
    return xa_weighted_average

def get_wghts(lat):
    """
    get latitude weights for gaussian grid.
    :param lat: latitudes
    :return: weights
    """
    latr = np.deg2rad(lat) # convert to radians
    weights = np.cos(latr) # calc weights
    return weights


path_a='temp/sea_ice_slice_vrs_11.nc'
make_folders.make_folders(path_a)
```

```
# Lat for NYA 79 Lon for NYA 12
# Lat for VRS 81 Lon for VRS -16
# We say plus minus 4 deg
sea_ice_slice11=sea_ice11.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_a)

path_b='temp/sea_ice_slice_vrs_12.nc'
make_folders.make_folders(path_b)
sea_ice_slice12=sea_ice12.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_b)

path_c='temp/sea_ice_slice_vrs_13.nc'
make_folders.make_folders(path_c)
sea_ice_slice13=sea_ice13.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_c)

path_d='temp/sea_ice_slice_vrs_14.nc'
make_folders.make_folders(path_d)
sea_ice_slice14=sea_ice14.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_d)

path11='temp/sea_ice_slice_vrs_11.nc'
path12='temp/sea_ice_slice_vrs_12.nc'
path13='temp/sea_ice_slice_vrs_13.nc'
path14='temp/sea_ice_slice_vrs_14.nc'

sea_ice_slice_wrking11 = xr.open_dataset(path)
sea_ice_slice_wrking12 = xr.open_dataset(path)
sea_ice_slice_wrking13 = xr.open_dataset(path)
sea_ice_slice_wrking14 = xr.open_dataset(path)

glob_mean_si11 = masked_average(sea_ice_slice_wrking11['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking11['lat']))
glob_mean_si11.to_netcdf('temp/glob_mean_si11_vrs.nc')

glob_mean_si12 = masked_average(sea_ice_slice_wrking12['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking12['lat']))
glob_mean_si12.to_netcdf('temp/glob_mean_si12_vrs.nc')

glob_mean_si13 = masked_average(sea_ice_slice_wrking13['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking13['lat']))
glob_mean_si13.to_netcdf('temp/glob_mean_si13_vrs.nc')

glob_mean_si14 = masked_average(sea_ice_slice_wrking14['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking14['lat']))
glob_mean_si14.to_netcdf('temp/glob_mean_si14_vrs.nc')
```

```
a=xr.open_dataset('temp/glob_mean_si11_vrs.nc')
b=xr.open_dataset('temp/glob_mean_si12_vrs.nc')
c=xr.open_dataset('temp/glob_mean_si13_vrs.nc')
d=xr.open_dataset('temp/glob_mean_si14_vrs.nc')
e=xr.merge((a,b))
f=xr.merge((e,c))
glob_mean_si_vrs_11_14=xr.merge((f,d))

glob_mean_si_vrs_11_14['SI']=glob_mean_si_vrs_11_14['__xarray_dataarray_variable__']
glob_mean_si_vrs_11_14.drop('__xarray_dataarray_variable__')

aaa = pd.DataFrame({'time': glob_mean_siNYA_11_14.time, 'SI':␣
 ↪glob_mean_siNYA_11_14.SI})

aaa.to_csv('AMDE_SI_vrs_11_14.csv')

path_a_nya11='temp/sea_ice_slice_nya_11.nc'
make_folders.make_folders(path_a_nya11)
# Lat for NYA 79 Lon for NYA 12
# Lat for VRS 81 Lon for VRS -16
# We say plus minus 4 deg
sea_ice_slice11nya=sea_ice11.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_a_nya11)

path_b_nya12='temp/sea_ice_slice_nya_12.nc'
make_folders.make_folders(path_b_nya12)
sea_ice_slice12nya=sea_ice12.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_b_nya12)

path_c_nya13='temp/sea_ice_slice_nya_13.nc'
make_folders.make_folders(path_c_nya13)
sea_ice_slice13nya=sea_ice13.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_c_nya13)

path_d_nya14='temp/sea_ice_slice_nya_14.nc'
make_folders.make_folders(path_d_nya14)
sea_ice_slice14nya=sea_ice14.sel(lat=slice(77, 85), lon=slice(340,348)).
 ↪to_netcdf(path_d_nya14)

path11nya='temp/sea_ice_slice_nya_11.nc'
path12nya='temp/sea_ice_slice_nya_12.nc'
path13nya='temp/sea_ice_slice_nya_13.nc'
path14nya='temp/sea_ice_slice_nya_14.nc'

sea_ice_slice_wrking11nya = xr.open_dataset(path11nya)
sea_ice_slice_wrking12nya = xr.open_dataset(path12nya)
```

```
sea_ice_slice_wrking13nya = xr.open_dataset(path13nya)
sea_ice_slice_wrking14nya = xr.open_dataset(path14nya)

glob_mean_si11nya = masked_average(sea_ice_slice_wrking11['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking11['lat']))
glob_mean_si11nya.to_netcdf('temp/glob_mean_si11_NYA.nc')

glob_mean_si12nya = masked_average(sea_ice_slice_wrking12['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking12['lat']))
glob_mean_si12nya.to_netcdf('temp/glob_mean_si12_NYA.nc')

glob_mean_si13nya = masked_average(sea_ice_slice_wrking13['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking13['lat']))
glob_mean_si13nya.to_netcdf('temp/glob_mean_si13_NYA.nc')

glob_mean_si14nya = masked_average(sea_ice_slice_wrking14['siconc'],␣
 ↪dim=['lat','lon'], weights=get_wghts(sea_ice_slice_wrking14['lat']))
glob_mean_si14nya.to_netcdf('temp/glob_mean_si14_NYA.nc')

a=xr.open_dataset('temp/glob_mean_si12_NYA.nc')
b=xr.open_dataset('temp/glob_mean_si13_NYA.nc')
c=xr.open_dataset('temp/glob_mean_si14_NYA.nc')
d=xr.open_dataset('temp/glob_mean_si11_NYA.nc')
e=xr.merge((a,b))
f=xr.merge((e,c))
glob_mean_siNYA_11_14=xr.merge((f,d))
glob_mean_siNYA_11_14

glob_mean_siNYA_11_14['SI']=glob_mean_siNYA_11_14['__xarray_dataarray_variable__']
glob_mean_siNYA_11_14.drop('__xarray_dataarray_variable__')

bbb = pd.DataFrame({'time': glob_mean_siNYA_11_14.time, 'SI':␣
 ↪glob_mean_siNYA_11_14.SI})

bbb.to_csv('AMDE_SI_NYA_11_14.csv')
```