**PMEL CCNC SOFTWARE MANUAL**

**AUGUST 14, 2017**

**Updated January 17, 2018**

**Updated February 23, 2018**

**Updated March 15, 2018**

**Operating the CCNC in polydisperse (spectra) mode**:

For ambient heated/unheated measurements, coordinate the timing of the thermal denuder valve with turning the CCN program on.

**Operating the CCNC in monodisperse mode**:

Make sure the SMPS is in "Analog" mode.

Start CCNC before the labview program so that there are multiple variables for lab view to pick up.

Once everything is running, make sure data are being saved to data gate by plotting the following in dchart:

ccn\_concentration\_ccn\_mono

ccn\_ss\_ccn\_mono

cn\_ccn\_mono

zi\_smps\_dp\_ccn\_mono

The first two variables indicate that the CCN data are being saved. The second two that the CN and diameter data are being saved.

For both the polydisperse and the monodisperse modes, keep dchart open so you can frequently check to make sure the data are saving and the instrument is operating properly.

**Standardized variable names for use on the wiki**

***Activity Headers***

Sea Sweep – monodisperse

MART – monodisperse

Ambient – polydisperse

Ambient behind denuder – polydisperse

Calibration – NaCl - monodisperse

Calibration – NaCl - polydisperse

Calibration – Amm Sulf - monodisperse

Calibration – Amm Sulf – polydisperse

Seawater – monodisperse

Seawater - polydisperse

Microlayer – monodisperse

Microlayer – polydisperse

***Lab Notebook Entry Type***

CCN

MART

Sea Sweep

TTHDMA

Calibration

Ambient

Seawater

Sea Sweep Blank

Microlayer

Monodisperse

Polydisperse (Spectra?)

**Downloading data from dchart**:

In dchart, select the variables below and the time range. Use separate browser windows for downloading and viewing data to avoid trying to download data while updating the time series display.

ccn\_concentration\_ccn\_mono (\_spectra)

ccn\_flow\_sample\_ccn\_mono (\_spectra)

ccn\_flow\_sheath\_ccn\_mono (\_spectra)

ccn\_ss\_ccn\_mono (\_spectra)

cn\_ccn\_mono (same variable name for spectra)

zi\_flag\_temp\_stable\_ccn\_mono (\_spectra)

zi\_smps\_dp\_ccn\_mono (not for spectra)

cn\_residual\_cpc\_tthdma (spectra only, not mono)

cn\_water

**Loading data into IGOR**:

Make two IGOR experiments - one for the mono data and one for the spectra (polydisperse) data.

Use the drop down menus in “ACG” to load data:

ACG Data, CCN, load data, from dchart\_mono (\_spectra)

If the data don't load, it usually means not all of the parameters listed above were downloaded. You can check this by double clicking on the .itx file to view it in IGOR. Don't do this when another experiment is open in IGOR, though, or you could overwrite existing variables.

***Loading valve flag (for denuder):***

Make a “valve” folder in the root directory.

Choose the valve folder as the Current Data Folder.

ACG – CCN – load data – tthdma

ACG – Utils - load data – load dchart itx file

**Uploading data and sample information to the ship’s ftp server**:

Upload IGOR .itx files along with the latest version of the CCN and AeroChem notebooks (.pdf format) to the ship’s ftp server. Try to do this once a day. I (Trish) will retrieve the data and look for problems, etc.

This is as far as you need to go. I will be looking at the data that you send.

But if you find time to do more, here are the next steps:

**Mono data processing**

ccn\_init\_all() //creates folders to put stuff in

ccn\_init\_sample\_info\_waves() //run once to create a data folder for flag waves

--got error: a wave read error: “Attempt to operate on a null (missing) wave” {this happened because data ended up in root folder, and program didn’t like it. Make sure data goes into ccn folder}

ccn\_update\_sample\_type\_dev(name,[active]) //adds a sample type to the sample type wave (e.g., ccn\_update\_sample\_type\_dev("AmmSulf\_031618"))

{I propose – Calibration, Ambient, MART, SeaSweep, Seawater, Microlayer}

ccn\_create\_cheat\_sheet(show="true") //opens cheat sheet tables with sample type and flag waves, use show ="true" to run the procedure automatically when adding a sample type or flag.

ccn\_update\_flag\_dev(name,[active,state\_list]) //adds a new flag to the flag wave (e.g. ccn\_update\_flag\_dev("MART"))

I propose-

TH\_DENUDER | UNHEATED;HEATED (built in)

AmSO4

NaCl

Instant Ocean

thru mast

Manifold

direct

pre-cruise

post-cruise

run 1

run 2

run 3

static

MART

flow-through

transit to station 1

Station 1

transit to station 2

Station 2

etc

ccn\_sample\_info\_plot() //creates plots to marquee for designating sample type periods

Marquee sample types

Once sample types have been created, make waves and matrix for the different sample types, %SS, and diameters: dev\_mono\_generate\_ss\_waves() // generates waves for all sample types

dev\_mono\_generate\_ss\_waves(sample\_type\_list="NaCl\_031718") //generates waves for ONLY specified sample types. With this option, only those sample types will be plotted using the functions below.

dev\_mono\_generate\_ss\_matrix() //generates the matrix using all sample types.

dev\_mono\_generate\_ss\_matrix([normalize]) //To normalize the supersaturation curves so

that the largest Dp CCN/CN ratio is 1 use (normalize=”true”) -- dev\_mono\_generate\_ss\_matrix(normalize="true").

Unnormalized data goes in to the root:ccn:dp\_v\_ss folder.

Normalized data goes into the root:ccn:dp\_v\_ss:normalize folder.

If normalized, use the dev\_mono\_plot\_activation as normalized.

Whenever you change or add sample types, the waves and matrix need to be generated using these two procedures.

**Mono data plotting**

dev\_mono\_plot\_activation(dp=50) //plots all identified sample types for dp = 50 nm

dev\_mono\_plot\_activation([dp,ss,normalize,sample\_type\_list,flag\_list]) //will plot for the

parameters designated, e.g., dev\_mono\_plot\_activation(dp=50,ss=0.1,normalize=”true”)

dev\_mono\_plot\_dp\_ss\_wave(50,0.1,sample\_type="AmmSulf\_031618") //will plot date vs. CN, CCN, and CCN/CN for specified diameters and %SS and sample types. Can be used to NAN periods of bad data. Shows what data points went into each diameter/SS pair in the dev\_mono\_plot\_dp\_ss\_wave plots. This is useful for figuring out "whacky" dp/SS pair data in the above plots. If data are obviously bad, you can remove it from the average in the above plots by marquee, right click, ccn\_mono: dp\_vs\_ss\_wave

NaN. If you decide later, that the data were actually good, redo the marquee and choose “reset”. Everytime you Nan in the time series plots, you need to renormalize the matrix (if you want it normalized to 1).

**Spectra data processing**

ccn\_init\_all() //creates folders to put stuff in

ccn\_init\_sample\_info\_waves() //run once to create a data folder for flag waves

ccn\_update\_sample\_type\_dev(name,[active]) //adds a sample type to the sample type wave (e.g., ccn\_update\_sample\_type\_dev("AmmSulf\_031618"))

ccn\_create\_cheat\_sheet(show="true") //opens cheat sheet tables with sample type and flag waves, use show ="true" to run the procedure automatically when adding a sample type or flag.

ccn\_update\_flag\_dev(name,[active,state\_list]) //adds a new flag to the flag wave (e.g. ccn\_update\_flag\_dev("MART")

ccn\_sample\_info\_plot() //creates plots to marquee for designating sample type periods

Marquee sample types

Once sample types have been created, make waves and the matrix for the different sample types, %SS, and diameters:

dev\_spectra\_generate\_ss\_waves()dev\_spectra\_generate\_ss\_matrix([normalize])dev\_spectra\_plot\_activation([normalize,sample\_type\_list,flag\_list])dev\_spectra\_plot\_activation([normalize,sample\_type\_list,flag\_list])

The generate\_waves function uses the waves below to calculate the ccn/cn ratio.

Waves used to calculate ratio:

CCN\_Concentration\_Cleaned

mono\_cn\_conc\_shifted

Calculating ss\_crit and Kappa based on mono dp\_v\_ss matrices for a given sample type list:

**FIND SS\_CRIT:**

Uses ccn\_kappa\_dev.ipf

Make sure the sample type list is correct in the ccn\_kappa\_dev.ipf.

ccn\_mono\_find\_ss\_crit(samp\_types,normalized) where samp\_types is ccn\_samp\_type\_list or another sample type list. For example:

ccn\_mono\_find\_ss\_crit(ccn\_samp\_type\_list, normalize = “yes”)

ccn\_mono\_find\_ss\_crit(ccn\_samp\_type\_list, normalize = “no”)

This function uses a spline fit to find the critical supersaturation for each sample type.

**PLOT SS\_CRIT:**

ccn\_mono\_plot\_ss\_crit(samp\_types,[normalize,error\_bars])

**FIND KAPPA:**

Uses ccn\_kappa\_dev.ipf

ccn\_mono\_find\_kappa(samp\_types,[normalize])

ccn\_mono\_find\_kappa(ccn\_samp\_type\_list,normalize = “yes”)

**PLOT KAPPA:**

ccn\_mono\_plot\_kappa(samp\_types,[normalize,error\_bars])

ccn\_mono\_plot\_kappa(ccn\_samp\_type\_list,normalize=”yes”, error\_bars=”yes”)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Calculating ss\_crit and kappa for mono sample types

ccn\_generate\_dp\_vs\_ss\_matrix(normalize="yes")

ccn\_generate\_dp\_vs\_ss\_matrix(normalize="no")

ccn\_mono\_find\_ss\_crit(ccn\_samp\_type\_list,normalize="no")

ccn\_mono\_find\_kappa(ccn\_samp\_type\_list,normalize="no")

ccn\_mono\_plot\_kappa(ccn\_samp\_type\_list,normalize ="no")

ccn\_mono\_plot\_ss\_crit(ccn\_samp\_type\_list,normalize ="no")

To find kappa from the comand line:

print ccn\_find\_kappa(.291, 50, 0.073)

To print standard kappa grid:

ccn\_make\_standard\_kappa\_plot()