CSEM integration

Surface object definition changes.

This entails a change in CRTM interface.

Separate land/water/snow/ice "subobjects" are needed to allow for multiples of each surface type to be defined in the input to the CRTM. Each individual subobject will be a CSEM base type and the CRTM versions will be a superclass of those. For example, a CSEM object definition would look like:

```
TYPE :: CSEM_Land_type
  INTEGER  :: Land_Type             = default value
  REAL(fp) :: Land_Temperature      = default value
  REAL(fp) :: Soil_Moisture_Content = default value
```

...etc...other components

```
CONTAINS
```

...CSEM surface type bound procedures

```
END TYPE CSEM_Land_type
```

similarly for water, snow, and ice. The subsequent CRTM superclass definitions would extend those, e.g.

```
TYPE, EXTENDS(CSEM_Land_type) :: CRTM_Land_type
```

...etc...CRTM-specific components

```
CONTAINS
```

...CRTM surface type bound procedures

```
END TYPE CRTM_Land_type
```

Again, similarly for water, snow, and ice. The main CRTM surface object would then become a container for the different gross surface types:

```
TYPE :: CRTM_Surface_type
  REAL(fp) :: Land_Coverage  = ZERO
  REAL(fp) :: Water_Coverage = ZERO
  REAL(fp) :: Snow_Coverage  = ZERO
  REAL(fp) :: Ice_Coverage   = ZERO
  TYPE(CRTM_Land_type),  ALLOCATABLE :: Land(:)
  TYPE(CRTM_Water_type), ALLOCATABLE :: Water(:)
  TYPE(CRTM_Snow_type),  ALLOCATABLE :: Snow(:)
  TYPE(CRTM_Ice_type),   ALLOCATABLE :: Ice(:)
  ...etc...other components
CONTAINS
  ....CRTM Surface type bound procedures
END TYPE CRTM_Surface_type
```

This is a big change to the current `CRTM_Surface_type` definition and will impact CRTM users in that they will need to modify their calling code.

The remaining questions:

1. To fully decouple the CSEM and CRTM surface type definitions, do we need Get/Set functions for the CSEM surface object definitions?
2. Similarly for the main CRTM Surface object definition, do we need Get/Set functions for the CRTM surface object definitions?

Internal SfcOptics procedure changes

This will involve changing the various surface optics ("SfcOptics") procedures in use in the CRTM to use those defined in the CSEM.

The CSEM SfcOptics procedure interfaces are similar to those currently used in the CRTM, but it's more than likely some refactoring of the CRTM procedures will be necessary.

This will involve changes to the main `CRTM_SfcOptics` module to call the CSEM forms of the various `Compute_`*spc-type*_*sfc-type*`_SfcOptics`[_*deriv-type*] procedures, where:

- *spc-type* is `MW`, `IR`, or `VIS`;
- *sfc-type* is `Land`, `Water`, `Snow`, or `Ice`; and
- *deriv-type* is none (forward model), `TL`, or `AD`.

The current main `CRTM_SfcOptics` module `USE` statements look like:

```
USE CRTM_MW_Water_SfcOptics, &
  ONLY: MWWSOVar_type => iVar_type   , &
        Compute_MW_Water_SfcOptics   , &
        Compute_MW_Water_SfcOptics_TL, &
        Compute_MW_Water_SfcOptics_AD
```

...etc...other microwave modules

```
USE CRTM_IR_Land_SfcOptics, &
  ONLY: IRLSOVar_type => iVar_type  , &
        Compute_IR_Land_SfcOptics   , &
        Compute_IR_Land_SfcOptics_TL, &
        Compute_IR_Land_SfcOptics_AD
```

...etc...other infrared modules

```
USE CRTM_VIS_Snow_SfcOptics, &
  ONLY: VISSSOVar_type => iVar_type  , &
        Compute_VIS_Snow_SfcOptics   , &
        Compute_VIS_Snow_SfcOptics_TL, &
        Compute_VIS_Snow_SfcOptics_AD
```

...etc...other visible modules

The interfaces to all of the SfcOptics procedures are constructed in a similarly common form since the inputs for most are the same. As such, replacing the various CRTM modules with the CSEM equivalents should mostly involve modifying the `USE` statements.

However, I expect that other changes will be required since Ming Chen has made improvements to the CSEM form of the SfcOptics procedures (different algorithms require additional inputs, bugs fixed, etc). At this point, I can’t comment as to the scope of those changes but I do not think they will be difficult to implement.