**Описание класса CAMBALA для расчета модовых функций и горизонтальных волновых чисел**

**Общие сведения**

1. Класс CAMBALA создан для организации доступа сторонних программ к модулю для расчета волновых k\_j чисел и собственных функций \phi\_j(z) акустической спектральной задачи.

2. Физическими входными параметрами для решения задачи являются: а) частота; б) шаг по глубине; в) массивы (векторы) данных, содержащие скорости звука, плотности и границы раздела (параметры среды); г) подмножество мод (критерий выделения – углы распространения).

3. Решение задачи зависит от ряда технических параметров: а) порядок экстраполяции Ричардсона; б) …

4. Выходными данными программы могут являться: а) волновые числа k\_j (индекс j пробегает конечный ряд значений сообразно подмножеству искомых мод, см. 2г); б) волновые числа и групповые скорости мод; в) волновые числа, групповые скорости мод, модовые функции. Кроме этого программа может выдавать г) модальные коэффициенты затухания (не реализовано в текущей версии CAMBALA), д) производные волновых чисел и модовых функций по глубине моря (не реализовано), е) …

**Инициализация объекта класса**

1. Объект класса может быть инициализирован двумя способами:

а) путем передачи всех требуемых параметров как входного аргумента;

б) путем чтения всех требуемых параметров их файла, сходного по структуре с файлом сценария геоакустической инверсии.

2. Необходимый набор входных данных соответствует аргументам функций

compute\_wnumbers\_extrap\_lin\_dz(), compute\_wnumbers\_extrap2() и т.п.. Он включает в себя:

а) частоту f в Гц (соотв. аргумент omeg = 2\*pi\*f);

б) vector<double> &depths – вектор глубин границ слоев;

в) vector<double> &c1s, vector<double> &c2s – вектор значений скорости звука на верхних и нижних границах каждого слоя (скорость звука внутри слоя считаем линейной функцией);

г) vector<double> &rhos – вектор значений плотности (плотность внутри каждого слоя считаем постоянной);

д) vector<unsigned> &Ns\_points – количества точек сетки в каждом слое;

е) double iModesSubset – подмножество мод.

Дополнительно: необходимо задавать горизонты для расчета модовых функций vector<double> &zr (см. ниже).

**Функции, которые необходимо запускать с помощью класса**

В настоящий момент класс можно использовать для запуска следующих функций из compute.h (набор будет расширяться, нужно описать механизм добавления функций)

1. функции для расчета волновых чисел (результат работы – горизонтальные волновые числа мод k\_j):

compute\_wnumbers\_extrap\_lin\_dz() (основная, запускается, если другое не затребовано настройками)

compute\_wnumbers\_extrap2 (вспомогательная)

compute\_wnumbers\_extrap (вспомогательная)

2. функции для расчета модовых функций (результат работы – одна или несколько модовых функция \phi\_j(z)) **\*внимание! все функции расчета мод требуют предварительного расчета волновых чисел\***

compute\_mfunctions\_zr() (основная, результат работы сохраняет в vector<vector<double>> &mfunctions\_zr, причем значения модовых функций считает на данном наборе горизонтов vector<double> &zr)

compute\_wmode() (вспомогательная, расчет одной модовой функции)

compute\_wmode1() (вспомогательная, расчет одной модовой функции)

3. функции для расчета групповых скоростей

… to appear

Вывод результатов

Результаты должны сохраняться в полях класса для последующего использования в других программах, где данный класс будет задействован. Кроме того, необходимо предусмотреть возможность вывода результатов в текстовый файл с разделителем в виде табуляции. Результаты расчетов следует сохранять в полях-переменных

1. &khs – волновые числа мод;

2. &mfunctions\_zr – модовые функции на данных горизонтах.

**О полях и методах класса**

Будем считать, что класс называется NormalModes. Объект данного класса представляет собой объединение характеристик некоторого вертикального сечения волновода мелкого моря, модовой структуры этого сечения и методов расчета отдельных его характеристик. **Рассмотрим поля класса.** Желательно, чтобы поля, соответствующие разным смысловым группам, как-либо отделялись друг от друга.

Поля переменных среды

Содержит c1s, c2s, rhos, depths и другие параметры среды (в дальнейшем они будут добавляться, например, будут добавлены затухания). Рекомендуемый префикс M (media).

Поля переменных источника/приемника

Содержит набор горизонтов zr (на которых будут вычисляться модовые функции), длину/длины трассы R (может быть вектором), частоту/частоты f (может быть вектором для расчета нестационарных полей).

Поля (вычисляемых!) волновых чисел и модовых функций

Содержат волновые числа khs, модовые функции на данных горизонтах mfunctions\_zr (в дальнейшем будут добавлены модальные коэффициенты затухания, производные волновых чисел и собственных функций по глубине моря и т.д.). Эти поля не заполняются пользователем/из файлов, а формируются в результате расчетов внутри объекта класса с помощью его методов.

**Рассмотрим методы класса.** Методы класса а) вводы/вывод данных (в файл или в программы, использующие класс); б) обеспечивают доступ к расчетным процедурам. Можно выделить следующие типы методов:

Ввод данных

Методы данной группы позволяют задавать поля переменных среды/параметров источника путем чтения из файлов (read) или задания из внешней программы (set). Методы второго типа могут быть использованы в конструкторе объекта класса (конструктор создает класс и принимает параметры среды в качестве аргументов).

Пример: set\_c1s(<vector<double>> c1s) – устанавливает значение поля c1s

read\_media(fname) – читает файл fname, в столбцах котрого стоят значения depths, c1s, c2s, rhos, затем заполяет считанными данными соотв. поля.

Вывод данных

Эта группа методов необходима для вывода результатов расчетов (волновых чисел, модовых функция и т.п.) в файлы (write) или во внешнюю программу (get). Во втором случае метод должен осуществлять проверку, произведено ли уже вычисление запрашиваемых величин (безопасный вариант по сравнению с простым обращением к полю).

Выполнение расчетов

Методы этой группы необходимы для вызова расчетных функций из compute.h. Они проверяют наличие/целостность необходимых данных (в полях объекта), затем вызывают расчетную подпрограмму, передавая ей значения полей в качестве аргументов и записывают результаты расчетов в соответствующие поля. У методов (в отличие от расчетных функций) практически не будет аргументов, т.к. они будут брать их из соотв. полей.

Пример: метод compute\_khs. Проверяет наличие данные в полях c1s, c2s, rhos, depths, f. В случае, если данные есть, вызывает compute\_wnumbers\_extrap\_lin\_dz(), передавая значения этих полей, результат записывает в поле khs.