Валидация программного обеспечения

Введение.

Проблемы, возникающие при использовании прикладных математических пакетов для численных расчётов в различных областях науки и техники известны давно, начиная с первых применений компьютеров. Они обусловлены (в конечном итоге) потерей точности вычислений, возникающей из-за применяемых математических алгоритмов, дискретности и конечности представления чисел в компьютере, адекватности математической модели, используемой программой. Так, например: из-за ошибки в математическом алгоритме был потерян спутник Mariner I (1962г.), из-за ошибок округления в программе, с помощью которой оценивалась дистанция до другого объекта, американский шаттл испытал трудности со встречей с космическим спутником Intelsat 6. Целая серия катастроф ракетно-космической, авиационной и ядерно-энергетической техники "катастрофический феномен 1985-1986 годов" [1] была вызвана неадекватностью математических моделей сложных объектов, систем их управления и программного обеспечения вероятностных расчётов.

В связи с этим, возникает задача «валидации» программы (или алгоритма, как части программы) путём исследования её свойств на моделях исходных данных. (Валидация, согласно ISO 9000:2005- это подтверждение посредством представления объективных свидетельств того, что требования, предназначенные для конкретного предполагаемого использования или применения, выполнены.) В нашей стране используются также понятия общей и метрологической аттестации. Следует отметить, что понятие «метрологическая аттестация» близко к используемому за рубежом понятию «валидация». Результатом общей аттестации программы служат оценки характеристик точности, устойчивости и сложности алгоритмов при различных моделях входных данных. Результатом же, метрологической аттестации являются оценки характеристик составляющих погрешности (неопределенности) результатов обработки в конкретных условиях применения этого алгоритма. Аттестуемая программа не должна вносить значимых погрешностей в суммарную погрешность результата, т.е. собственные погрешности программы должны быть малы по сравнению с погрешностями входных данных и методическими погрешностями алгоритма. Рекомендации по аттестации программ (алгоритмов) изложены в МИ 2174-91, МИ 2955 – 2010, МИ 2174, Р 50.2.077-2011,
OIML D 31, Welmec 7.2.

Как правило, для обработки результатов измерений широко применяются распространённые коммерческие прикладные программные пакеты: Microsoft Excel, MathSoft MathCad и MathWorks Matlab. Как показали проведенные исследования [2], в определённых случаях программа Microsoft Excel даёт ошибочные значения для некоторых статистических функций. Были проведены также исследования таких пакетов, как: MathSoft MathCad и MathWorks Matlab [3-5].

В настоящий момент бурно развивается бесплатное, некоммерческое программное обеспечение, которое возможно использовать в метрологической практике. Использование некоммерческого программного обеспечения с открытым кодом позволяет применить также аналитические методы при исследовании функциональных свойств ПО, оценить его влияния на точность конечного результата и повысить достоверность результатов исследования. Среди таких программ видное место занимает свободно распространяемый математический пакет с открытым кодом, как Scilab.

Методы оценивания параметров точности ПО.

Рассмотрим источники возникновения ошибок и погрешностей при расчётах.

При решении задачи с помощью вычислительной техники существует целый ряд этапов:

1.) Построение математической модели,

2.) Разработка или выбор численного метода,

3.) Разработка алгоритма,

4.) Программирование,

5.) Проведение вычислений.

Каждый из этих этапов может быть источником погрешностей, влияя тем самым на достоверность (точность) окончательного результата. Например, исходные экспериментальные или расчётные данные задачи часто являются основным источником погрешностей, математическая модель также вносит свой вклад в погрешность результата из-за того, что она является лишь приближенным описанием реального процесса или явления. Такие погрешности называются *неустранимыми*, т.е. в ходе последующих вычислений их нельзя устранить.

При использовании численного метода для решения математической задачи, ещё не приступив к вычислениям, уже допускается новая погрешность, которая называется *погрешностью метода*. Она связана с тем, что любой численный метод описывает исходную математическую модель приближенно. Типичные погрешности метода - погрешность дискретизации и погрешность округления. При численном решении математической задачи на компьютере создают её дискретную модель. Тогда, разность решений исходной и дискретизированной задачи называется *погрешностью дискретизации*. Погрешность дискретизации возникает из-за того, что при решении системы большого числа алгебраических уравнений, которая представляет собой дискретную модель, входные данные этой системы (коэффициенты и правые части) задаются в ЭВМ не точно, а с округлением. В процессе работы алгоритма погрешности округления накапливаются, и как следствие, решение, полученное компьютером, будет отличаться от точного решения дискретизованной задачи. Результирующая погрешность называется *погрешностью округления* (*вычислительной погрешностью*). Следующие факторы влияют на величину этой погрешности: точность представления вещественных чисел в компьютере и чувствительность использованного алгоритма к погрешностям округления. Эта погрешность теоретически может быть уменьшена до любого значения, но на практике её величину ограничивают до величины, в несколько раз меньшей неустранимой погрешности. Потому, что повышение точности метода не приводит к повышению точности окончательного результата.

Помимо вышеизложенных причин, существуют и другие источники ошибок при применении компьютеров для вычислений. Ошибки, возникающие при расчётах можно условно разбить на три группы [8]:

1.) Ошибки, связанные с погрешностью, вносимой ПО при обработке данных, т.е. ошибками выбора и реализации алгоритмов обработки данных

2.) Ошибки, связанные с преднамеренным или неумышленным искажением исходного кода ПО

3.) Ошибки, связанные с искажением данных при их передаче, хранении и представлении.

На рисунке №1 представлены основные источники погрешностей, возникающие при обработке и преобразовании результатов измерений.

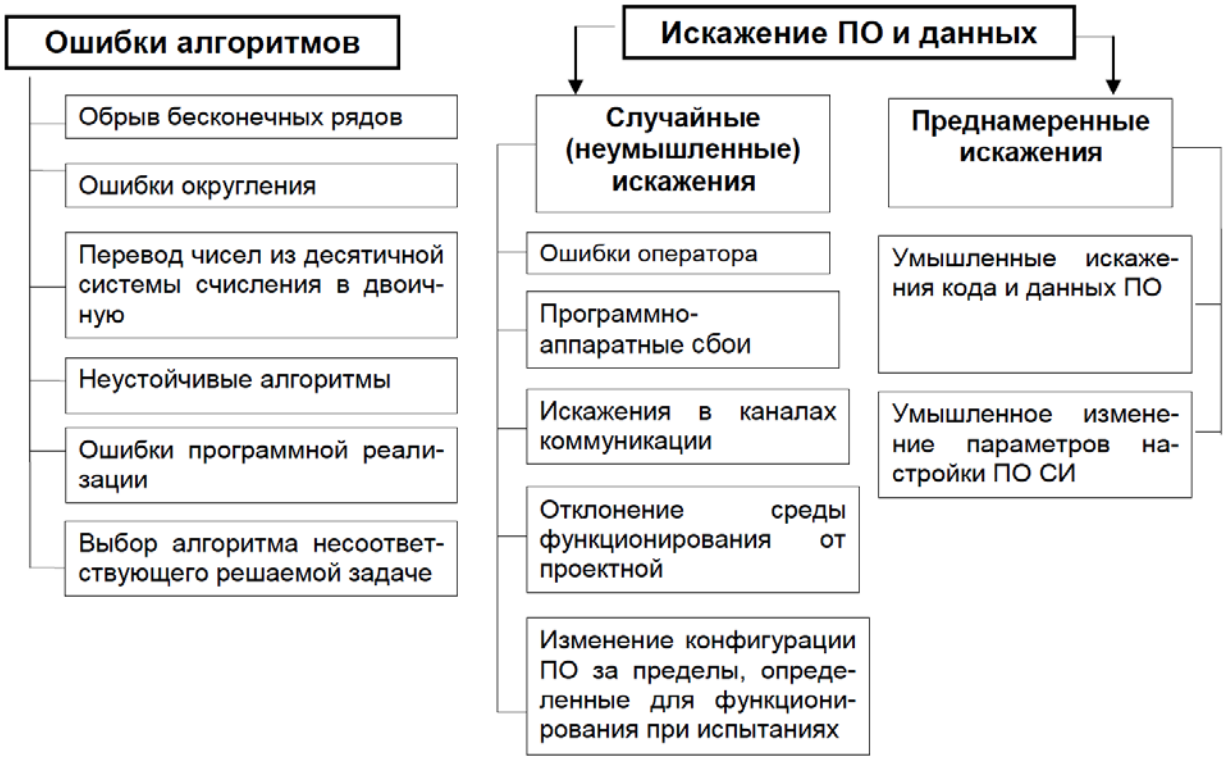

Рис.1 Источники ошибок ПО СИ [8]

Как было отмечено выше, одним из основных источников вычислительных погрешностей является приближенное представление чисел в компьютере из-за конечности разрядной сетки [9]. Кроме того, при вычислениях с плавающей точкой операция округления может потребоваться также после выполнения любой из арифметических операций. Т.е. число a подвергается округлению, и заменяется близким числом $\tilde{a}$, представимым в компьютере точно. Поэтому оценим границу относительной погрешности при представлении числа *a* в компьютере числом с плавающей точкой [10]. Используем правило округления – отбрасывание всех разрядов числа, которые выходят за пределы разрядной сетки. Система счисления – двоичная. Запишем число, представляющее бесконечную двоичную дробь

$$a = \underbrace{\pm 2^{p}}_{order} \underbrace{\left(\frac{a_1}{2}+\frac{a_2}{2^2}+\cdots+\frac{a_i}{2^i}+\frac{a_{i+1}}{2^{i+1}}+\cdots\right)}_{mantissa},$$

где $a_j = \begin{cases} 0, \\ 1, \end{cases} \quad (j = 1,2,\ldots)$ - цифры мантиссы.

Пусть под запись мантиссы отводится *t* двоичных разрядов. Отбросим лишние разряды, и получим округлённое число:

$$\tilde{a} = \pm 2^{p}\left(\frac{a_1}{2}+\frac{a_2}{2^2}+\cdots+\frac{a_i}{2^i}\right).$$

Тогда абсолютная погрешность округления будет равна

$$a-\tilde{a} = \pm 2^{p}\left(\frac{a_{i+1}}{2^{i+1}}+\frac{a_{i+2}}{2^{i+2}}+\cdots\right).$$

Наибольшая погрешность будет если выполняются условия:

$$a_{t+1}=1,\ \ a_{t+2}=1,$$
$$|a-\tilde{a}|\le \pm 2^{p}\frac{1}{2^{t+1}}\underbrace{\left(1+\frac{1}{2}+\frac{1}{2^2}+\cdots\right)}_{=2}=2^{p-t}.$$

Поскольку величина мантиссы M числа a $|M|\ge 0{,}5$, то всегда a_1=1.

Тогда $|a|\ge 2^{p}\cdot 2^{-1}=2^{p-1}$ и относительная погрешность равна $\frac{|a-\tilde{a}|}{|a|}\le 2^{-t+1}$. На практике используют более точные методы округления. В этом случае погрешность представления чисел равна

$$\frac{|a-\tilde{a}|}{|a|}\le 2^{-t},$$

т.е. точность представления чисел определяется разрядностью мантиссы t, и приближенно представленное в компьютере число можно записать в виде $\tilde{a}=a(1\pm\epsilon)$, где $|\epsilon|\le 2^{-t}$ – *"машинный эпсилон"* – относительная погрешность представления чисел.

Методы тестирования программного обеспечения

Тестирование ПО предназначено для проверки правильности его функционирования. В случае использования ПО для решения метрологических задач это означает проверку его пригодности для получения результата с требуемой точностью, т.е. рассматривается задача определения точностных характеристик результата измерения, получаемого с помощью тестируемого ПО.

Поскольку, чаще всего у используемого программного обеспечения (например: коммерческого) отсутствует описание конкретных методов программной реализации алгоритмов, то при проведении тестирования его часто представляют в виде «черного (или непрозрачного) ящика». В зависимости от наличия информации об используемых алгоритмах ПО можно представлять как «серый (или полупрозрачный) ящик», где «прозрачность» определяется степенью наличия такой информации. В идеале видится полностью «прозрачный» «ящик». Это даёт возможности использования аналитических методов при исследовании функциональных свойств ПО, оценки его влияния на точность конечного результата, и повышения достоверности результатов исследования. Использование ПО с открытым кодом даёт такую возможность, но реализация анализа исходного кода подразумевает огромный объём работы, что значительно усложняет задачу по его исследованию. Только в особо ответственных случаях используют анализ исходного кода ПО. Поэтому, наиболее распространённым методом тестирования ПО является метод «черного ящика» (black box testing). В этом методе сопоставляются результаты обработки «эталонных» данных, полученные тестируемым ПО и результаты,

полученных обработкой тех же данных «эталонным» или "опорным" ПО (рис.2).

Рис.2 Тестирование ПО с использованием опорного "эталонного" программного продукта

В случае отсутствия опорного («эталонного») ПО, сравнительные испытания проводят с использованием *моделей исходных данных* (МИ 2174 [17]), либо с применением *метода генерации «эталонных» данных* (англ. reference data set) [10] . При наличии нескольких ПО сопоставимого уровня и в отсутствии опорного («эталонного») ПО проводят сличения результатов, полученных с помощью этих программ (подобные сличению эталонных СИ), либо испытания на основе анализа исходного кода ПО, и комбинации указанных методов.

Схема тестирования ПО с использованием генерации “эталонных данных” представлена на рис. 3. На рис.4 представлена схема тестирования методом моделей исходных данных. Сравнивая эти две схемы, можно сделать вывод, что метод моделей исходных данных является частным случаем метода генерации “эталонных данных”.

Рис.3 Тестирование ПО с использованием генерации "эталонных данных"

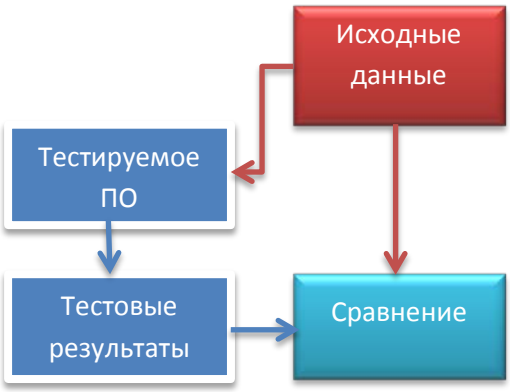

Рис.4 Тестирование ПО методом моделей исходных данных

Особенностью схемы, представленной на рис.3, является то, что на её вход подаются не «эталонные» данные, а «эталонные» результаты. Совокупность «эталонных» данных и «эталонных» результатов в работах NPL называют «эталонными» парами. Стоит отметить, что генерация «эталонных» данных по известным «эталонным» результатам не является однозначной (приходиться решать обратную задачу). Поэтому, в результате такой генерации получаются целые семейства или классы «эталонных» данных, которые удовлетворяют условиям математической задачи.

Перечислим требования, которые необходимы при тестировании ПО методом генерации "эталонных" данных [11]:

1.) «Эталонные» входные и выходные данные (может потребоваться программа генерирования «эталонных пар», которая должна быть также аттестована);

2.) «Эталонная» программа вычисления результата измерения;

3.) Программа обработки массивов выходных данных «эталонного» и тестируемого ПО для получения в общем случае закона распределения выходной величины и его характеристик: среднего и СКО.

Сами "эталонные" данные должны обеспечивать:

1.) Выявление дефектов в тестируемом ПО;

2.) Конструирование тестовых наборов с наперед известными свойствами (например, степень зашумленности);

3.) Возможность получения таких тестовых наборов входных данных, для которых «эталонные» выходы известны и которые, в некотором смысле, «близки» к наборам входных данных, возникающим в процессе практического применения ПО;

4.) Возможность получения большого количества тестовых наборов, гарантирующего достаточный охват возможных областей изменения входов программного обеспечения.

Такой подход позволяет проконтролировать корректность применяемого ПО. Для количественной оценки влияния ПО на точность результата вводятся метрики, с помощью которых можно оценить потерю точности, используя «эталонные» данные или результаты, получаемые «эталонным» ПО.

1.) Абсолютные меры точности.

Пусть

$$\Delta y = y^{(test)} - y^{(ref)},$$

где y(test) — тестовые выходные результаты, а y(ref) — «эталонные» выходные результаты, отвечающие «эталонному» входному n –мерному вектору x.

Тогда величина

$$d(x) = \| \Delta y \| / \sqrt{n}$$

является абсолютной мерой отклонения тестового результата от «эталонного» при «эталонном» входном векторе x.

2.) Относительные меры точности.

Пусть M(x) - число точных значащих цифр в «эталонных» результатах, отвечающих «эталонному» входному вектору x . Тогда формула для вычисления числа совпадающих цифр в результатах тестовых расчетов и «эталонных» результатах имеет вид:

$$N(x) = \min\left\{ M(x), \log_{10}\left(1 + \frac{\| y^{(ref)} \|}{\| \Delta y \|} \right) \right\}, \text{ если } M(x) \neq 0,$$

и N(x) = M(x) — в противном случае.

2.) Исполнительная характеристика — это величина

$$P(x) = \log_{10}\left(1 + \frac{1}{\kappa(x)\,\eta} \frac{\| y^{(ref)} \|}{\| \Delta y \|} \right),$$

где η - вычислительная точность; κ (x) — коэффициент обусловленности задачи. Коэффициент обусловленности вычисляется, как частное от деления относительного изменения выхода на относительное изменение входа:

$$\kappa(x) = \frac{\|\delta y\|}{\|y\|} \Big/ \frac{\|\delta x\|}{\|x\|}.$$

Исполнительная характеристика показывает число точных значащих цифр, «потерянных» в результате тестовых расчетов по сравнению с результатами, полученными программой, реализующей оптимально устойчивый алгоритм. В случае «эталонного» ПО имеем, что

$$\delta y \approx J(x)\delta x \text{, где } J(x) = \left\{\partial f_i / \partial x_j\right\}$$

J(x)- якобиан функции f(x).

Таким образом, «эталонное» ПО рассматривается, как полностью «прозрачный» ящик, для которого зависимость выходных данных от входных описывается функцией f(x). Дополнительные погрешности, обусловленные «эталонным» ПО, — это только погрешности округления

$$\|\Delta x\| \approx \eta \|x\|$$

где η — относительная точность. Тогда

$$\|\Delta y_{ref}\| = J(x)\|\Delta x\| = J(x)\eta_{ref}\|x_{ref}\| = k(x)\eta_{ref}\|y_{ref}\|$$

В случае, если тестирование ПО выполнялось с помощью «эталонных» данных (входных и выходных величин) без использования «эталонного» ПО, то используя, известную машинную точность тестируемого ПО можно аналитически оценить предельную точность выходных данных:

$$\|\Delta y_{\lim}\| = J(x)\|\Delta x\| = J(x)\eta_{test}\|x_{ref}\| = k(x)\eta_{test}\|y_{ref}\|.$$

Сравнив эту величину с реально наблюдаемыми отклонениями выходных данных тестируемого ПО и с «эталонными» выходными данными, можно оценить потерянную точность вследствие реализации алгоритма тестируемым программным обеспечением:

$$\frac{\|\Delta y\|}{\|\Delta y_{\lim}\|} = \frac{\|\Delta y\|}{k(x)\eta_{test}\|y_{ref}\|}.$$

В международных документах по валидации (аттестации) ПО такие характеристики называют «мерами поведения (функционирования)» ПО. Их используют для того, чтобы сделать заключение о пригодности тестируемого ПО для решения конкретной измерительной задачи.

Существует сайты, посвященные вопросам аттестации научного программного обеспечения. http://www.eurometros.org/ http://www.itl.nist.gov/div898/strd/index.html, http://www.npl.co.uk/science-technology/mathematics-modelling-and-simulation/mathematics-and-modelling-for-metrology/. На этих сайтах можно найти "эталонные данные", которые можно использовать для тестирования ПО.

1.) На европейском сайте EUROMETROS – размещаются математическое и статистическое программное обеспечение для метрологии, а также тестовые "эталонные" данные.

METROS (METROlogy Software) система была развита в рамках SSfM Software в Национальной физической лаборатории (Англия) (1998-2001) и описана в руководстве "The Guide to EUROMETROS: SSfM Good Practice Guide No. 5". В дальнейшем METROS стала развиваться как общеевропейская система и была переименована в EUROMETROS (2004-2007). Эта система рассматривается, как репозиторий (хранилище, где хранятся и поддерживаются какие-либо данные, чаще всего в виде файлов, доступных для дальнейшего распространения по сети) в которой хранится библиотека ключевых математических функций и алгоритмов, используемых в метрологии и разбитых по областям измерений, в которых они находят наибольшее применение. На сайте также находятся описания алгоритмов, реализующих эти функции.

2.) NIST Statistical Reference Datasets

Подразделение статистических инженерных и математических вычислительных методов американского института стандартов и технологий (NIST) http://www.itl.nist.gov/div898/strd/index.html поддерживает наборы сертифицированных эталонных тестов, которые называются Statistical Reference Datasets (StRD). Здесь представлены «эталонные» данные и соответствующие им «эталонные» результаты по следующим четырем основным наборам алгоритмов:

- дисперсионный анализ;

- линейная регрессия;

- нелинейная регрессия;

- одномерная суммарная статистика.

Для каждого набора алгоритмов «эталонные» наборы делятся по уровню сложности, классификации модели (например, экспоненциальная, рациональная или линейная, квадратическая, полиномиальная и др.).

Огромное количество пользователей используют StRD для проверки точности статистического софта, как коммерческого, так и бесплатного. Чтобы провести тестирование ПО с StRD, необходимо загрузить представленные данные в программу, запустить нужный анализ и сравнить результаты с сертифицированными величинами. Для каждого тестируемого алгоритма StRD содержит несколько уровней сложности, которые позволяют проверить различные аспекты алгоритма. Различают 3 уровня: низкий, средний и высокий. Степень точности алгоритма, в основном, соответствует этим трём уровням. Причём каждый из предложенных сетов содержит набор данных, полученных при решении реальной задачи. И для этих данных сертифицированные величины вычислялись с использованием суперкомпьютера с использованием языка FORTRAN с очень высоким уровнем точности (сертифицированные значения величин округлялись только после 15 значащей цифры).

Другой возможностью создания наборов "эталонных" данных является метод «нуль-пространства», предложенный в документах Национальной физической лаборатории (Англия) http://www.npl.co.uk/ . Этот метод позволяет любому пользователю ПО создать свой набор "эталонных" данных для тестирования ПО. Рассмотрим идеи и алгоритм, лежащий в основе этого метода.

Основная идея этого алгоритма заключается в том, что входные «эталонные» данные формируются таким образом, чтобы «эталонные» выходные данные оставались неизменными — так называемый метод «нуль-пространства». В этом случае различным наборам (векторам) входных «эталонных» данных соответствует единственный выходной набор (вектор) «эталонных» данных. Рассмотрим его применение на примере задачи линейной регрессии. Предположим, что решается задача линейной регрессии: задан вектор

$$y = \begin{bmatrix} y_1 & y_2 & \dots & y_m \end{bmatrix}^T$$

результатов наблюдений, проводившихся в дискретные моменты времени x_i

$i = \overline{1,m}$, при этом считаем, что результаты y_i определяются линейной зависимостью $y_i = b_1 + b_2 x_i + r_i$, $i = \overline{1,m}$, где $b_{1,2}$ — параметры, подлежащие оценке, r_i — случайные ошибки, возникающие в процессе измерения. Перепишем систему уравнений в матричной форме:

$$y = Ab + r, \tag{2.2}$$

где A — матрица наблюдений, y — вектор результатов наблюдения, b — вектор параметров модели, r — вектор остатков;

$$A = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_m \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{bmatrix}, \quad r = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \end{bmatrix}, \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}.$$

Известно, что решение задачи линейной регрессии по методу наименьших квадратов характеризуется тем, что

$$A^T r = 0.$$

Следовательно,

$$\sum_{i=1}^{m} r_i = 0, \quad \sum_{i=1}^{m} x_i r_i = 0.$$

Пусть N — матрица, столбцы которой являются базисными векторами нуль-пространства матрицы A^T, тогда $A^T N = 0$. Пусть $r = Nu$, тогда для различных векторов результатов наблюдения y и $y + r$ получаем одинаковые значения параметров b_i. Действительно, $A^T Ab = A^T y$ и $A^T (y + r) = A^T y + A^T Nu = A^T y$.

Распишем этапы построения тестовых наборов данных для данной задачи:

1. Вычисляется вектор результатов наблюдения $y_0 = Ab$;

2. Строится матрица N, столбцы которой представляют собой базис нуль-пространства для матрицы A^T ;

3. Формируется вектор остатков r = N u, где элементы вектора u генерируются при помощи датчика случайных чисел;

4. Проводится нормировка вектора r таким образом, чтобы он и его компоненты имели распределение с заданным средним значением и СКО;

5. Формируется вектор результатов наблюдения y в соответствии с формулой $y = y_0 + r$.

В заключение можно сказать, что целью таких тестов является проверка выбранных математических (статистических) функций конкретного программного продукта для определения степени точности расчетов в различных диапазонах значений входных данных и исполнительных параметров. В случае отсутствие «эталонного» программного обеспечения описанные методы тестирования могут быть использованы для оценки качества алгоритмов обработки измерительной информации в ПО средств измерений и измерительных систем.

Литература.

1. С.Ф. Левин Катастрофический феномен 1985-1986 годов: расчёты точности. Контрольно-измерительные приборы и системы №3 2013г.

2. H.R. Cook, M.G. Cox, M.P. Dainton, P.H. Harris. Methodology for testing spreadsheets and other packages used in metrology. Report to National Measurement System Policy Unit, September 1999

3. A.J. Cox, N.J. Higham. Accuracy and Stability of the Null Space Method for Solving the Equality Constrained Least Squares Problem, SIAM, 1998

4. M.G. Cox, M.P. Dainton, P.M. Harris. Testing Spreadsheets and Other Packages Used in Metrology. Testing Functions for the Calculation of the Standard Deviation. Report to National Measurement System Policy Unit. October 2000

5. M.G. Cox, M.P. Dainton, P.M. Harris. Testing Spreadsheets and Other Packages Used in Metrology. Testing Functions for the Linear Regression. Report to National Measurement System Policy Unit. October 2000

6. M. G. Cox, P. M. Harris. The design and use of reference data sets for testing scientific

7.Ю.А. Кудеяров, А.А. Сатановский ГЕНЕРАЦИЯ ЭТАЛОННЫХ ДАННЫХ МЕТОДОМ НУЛЬ-ПРОСТРАНСТВА ДЛЯ ТЕСТИРОВАНИЯ ЭЛЕКТРОННЫХ ТАБЛИЦ, ПРИКЛАДНЫХ МАТЕМАТИЧЕСКИХ ПАКЕТОВ И АЛГОРИТМОВ.

8. САТАНОВСКИЙ А. А. Дисс. к.т.н РАЗРАБОТКА НАУЧНО-МЕТОДИЧЕСКИХ ОСНОВ МЕТРОЛОГИЧЕСКОЙ АТТЕСТАЦИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ СРЕДСТВ ИЗМЕРЕНИЙ ФГУП «ВНИИМС» 2007г.

9.Международный стандарт представления чисел с плавающей точкой в ЭВМ- IEEE 754

10. http://www.machinelearning.ru/wiki/index.php?title=%D0%9E%D1%88%D0%B8%D0%B1%D0%BA%D0%B8_%D0%B2%D1%8B%D1%87%D0%B8%D1%81%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9

11. Слаев В.А., Чуновкина А.Г. Аттестация программного обеспечения, используемого в метрологии. Справочная книга./Под ред В.А. Слаева – С.Пб.: «Профессионал», 2009, 320 с.