Document Number: P2030R0
Date: 2020-01-13
Authors: Michael Wong
Project: Programming Language C++, SG19 Machine Learning
Reply to: Michael Wong <michael@codeplay.com>

SG19: Machine Learning 2019/10/10-2020/01/09

Contents

Minutes for 2019/10/10 SG19 Conference Call

>
> 1.1 Roll call of participants
>
Millan Girkar (Intel), Phil Ratsloff(SAS), Marco Foco (nvidia0, Andrew
Lumsdaine (U W), Richard Dosselwamn (U R), Michael Wong (CP), Jens Maurer,
Domagoj Saric

1.2 Adopt agenda
>
> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
> All C++ reflector are now moved to listserv
>
> https://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> 2.2 Paper reviews
>
> Review Belfast paper submission strategy
>
https://isocpp.org/papers
to submit papers
need an isocpp password

> 2.2.1: ML topics
> Differentiable Programing by Marco Foco
>
new person implemented in clang
He will be at Belfast
Vassil Vassilev

> Richard Dosselman
>
> P1708R1: Math proposal for Machine Learning
>
>
https://docs.google.com/document/d/1VAgcyvL1riMdGz7tQIT9eTtSSfV3CoCEMWKk8GvVuF

Y/edit
>
instead of independent functions that ake one pass over the data, that was based on python
we have a range where we can walk once so computation are free and do multiple scan
load a variety of entity over that range
and individually load the value to be evaluated

Mean: has an overload and not just set
just like evrything else in general standards algorithm template, follows the accumulator function as well
e.g:
integer list of fixed pre-determiend size
then accumulator which operates over that set
this is like boost accumulator library, given a set then load a function, from eric niebler
versatile as to which set it applies to
first one does just one statistics

e.g. 2
structure, o have to have notion of what mean is
now accumumulating over a set of point, but result is a single scalar
lambda describe what we mean to be computing the average
can use this to create midpoint, geometric and harmonic mean is for the future
3.3
median requires sorting you wont know the middle until sorted
quick select is a popular same designer as quick sort
return value: iterator to the individual mean median, now aming for structure where we return a first or second median
can ask for both mean and median
should we allow people to know there are 2 medians
why get<1> instead <0>, off by 1 error

when I apply median will get 2 values, magenta and cyan
python for median of even number integers, probably averages them resulting in floats as they dont have support for integers
no support for strings? averaging signals, images
dont design it to force string to work, it is outside of problem domains, but if it works great
for mean yo usum individual pixels and divide by number of elements
if you have an image unit8, when u accumulate it in a wider type, after dividing it goes back to unit8
so intermediate type is also different

std::accumulate already support different types for accumulator, this is
good

Mode: needs a hash table
where is the vector and why optional? possibility if there is no return
value at all
are there several results?
worry we are returning a vector as it allocates memory which means passing
in an allocator, a range would be better

but this would deviate from standard unlike median mean
we seem to have this problem anyway
Not sure how a range would be use in this case, it would have to be sorted
order

Standard deviation
simpler cases
dont do example 2 in 3.5, look at ranges for projections that have
algorithm built in
3.6 Variance
asked for by Jens,

Median /mode seems to need different treatment , needs memory allocation,
and hash tables

>
> P1709R1: Graph Proposal for Machine Learning
>
>
>
https://docs.google.com/document/d/1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE
/edit?usp=sharing
>
<https://nam02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.google.com%2Fd
ocument%2Fd%2F1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE%2Fedit%3Fusp
%3Dsharing&data=02%7C01%7CPhil.Ratzloff%40sas.com%7C729b2cf8502641e4ae5e08d749
064578%7Cb1c14d5c362545b3a4309552373a0c2f%7C0%7C0%7C637058163592253027&sdat
a=4UQm8tqrcUbiZsr200UMrOaEModJYGNgP1oNot9PbAg%3D&reserved=0>
>
>
>
was trying to support OO interface, now switching to a functional interface
like erasing an edge on a vertex, so need the context of a graph
store a graph pointer, so now start lookign at concepts, what are those
functions instead of having a table

COncepts: there are just definitions, no implemenation yet, trying to present the concepts that will be useful throughout definiiton we will be using
using _c to designate a concept type
have to get vertexes and edges of the graph
there is an arithmetic concept, based on the arithmetic type
for depth-search first

intoduced type_traits for is this an adjancey array, list , etc
there is graph value type
vertex key, vertex value,

ways of creating vertex, can pass vertex by rvalue reference, can erase vertexes or in a range, clear, find
can erase edges, or based on in out edge iterator
can find an edge based on vertex iterators or keys
can do same for in out edge
why does adjancey list return a Grpah type? Yes, it was suppose to be a pointer to an object (that is newly allocated object, then it needs to be a managed pointer or object by value, and should not be return by raw pointer; Ok if we return unique pointer the need to do more work for a shraed pointer) , need to specify what G means, this needs some more attention

why shoudl this helper function be at the top level? Should we not use concept based,
all kinds of data structure that can meet the concept of the graph or adjanceny list and that should make up a list of vectors, we had that in BGL; seems like you want to be able build up from that to sparse matrix

depth first and breath first search are really iterators! agree
so Iterator example shows how to create a graph G1 with adjacency list
DFS range first item is vertex where you are at, the second part s a sentinel
yes this makes perfect sense to me, definitely approach we have been using for BGL 17
same example in breath first search with different ordering coming out

graph_c G is reused everywehere, should they not be differerent concepts, as BFS and DFS is just reading, if you are reusing this for erase function
,
yes so far these are generic

TopoSort

Algorithm
shortest path
BFSP
dijkstra reuires nonnegatve weighs, consier a concept for that which would require unsigned integers
there are no unsigned doubles, and doubles is a possible weights
connected component is an undirected graph

Dont do tuples, as they dont have names as they use get<0>, which we dont know what they are
get named members like ranges
Do you have a description of graph_c concept? Yes in Concepts, directed and undirected
need to have good definitions
an unidrected graph should not be required to have an erase function. Is that from easable? May be just check
I Have a concept Erasable
could be mixing 2 things, algo which operate on a graph, and these Graph functions that appear to operate on predefined graph data structure that you offer in your library and they look the same but take your specific Graph type, they should take a generic graph type.
Agree, but they could be overwritten wit user-defined functions and these perform the desire goal.
But now you require each of these functions and that expands the interface greatly
I see if I want to adapt my thing, then I want to know what I want to override, this is the point of Concept
Right, I cant tell what the minimum set of operations that is needed

industry is moving away from library solutions like TVM, Glow these are compile time generated graphs
yes in scope just not today,
vertexes and edges are compield time constnat and these would be optimized

vertex iterator concept, some produce pairs of vertex iterator, no implied navigation on these vertexes
what does increment do on these? Nothing for shortest path. Perhaps minimum requirement to satisfy
so no need for ++ unless I really need it but for some algo, being an iterator vs being a pointer or reference really means, you dont really need a random access iterator into the next function?

example for transitive closure for modifying tht grah nd how to produce a fresh graph tht is a transitive closure of another graph to demonstrate the interface as given works why create_adjancency list just creates a graph. So why need a special creation function for graph? Because I was trying to create a functional definition.

adjacency list that is initiliazed with an initialized list? Yes useful

also having global functions that modify graphs.

P1416R1: SG19 - Linear Algebra for Data Science and Machine Learning
>
> https://docs.google.com/document/d/1IKUNiUhBgRURW-UkspK7fAAyIhfXuMxjk7xKikK4Yp8/edit#heading=h.tj9hitg7dbtr
>
> P1415: Machine Learning Layered list
>
>
https://docs.google.com/document/d/1elNFdIXWoetbxjO1OKol_Wj8fyi4Z4hogfj5tLVSj64/edit#heading=h.tj9hitg7dbtr
>
> 2.2.2 SG14 Linear Algebra progress:
> Different layers of proposal
>
>
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit
>
> 2.2.3 any other proposal for reviews?
>
> 2.3 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> - *2019-11-04 to 09: Belfast, Northern Ireland;* Archer Yates
>
> -2020-02-10 to 15: Prague, Czech Republic
>
> - 2020-06-01 to 06: Bulgaria
> - 2020-11: (New York, tentative)
> - 2021-02-22 to 27: Kona, HI, USA
>
> 3. Any other business
>
> New reflector

>
> http://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> Old Reflector
> https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19
> <https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg14>
>
> Code and proposal Staging area
>
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda
>
> TBD
>
> 5.2 Future meeting
>
> Aug 8
> Sep 12
> Oct 10: Mailing deadline Oct 7
> Nov 14 - cancelled due to DST change and switching to a new cycle.

Minutes for 2019/12/12 SG19 Conference Call

> 1.1 Roll call of participants
>
Michael Wong, Phil Ratzloff, Andrew Lumsdaine, Duygu Cakmak, Matthew
galati, Richard Dosselmann, Lukasz Wojakowski, Jesun Firoz, Scott McMillan,
DOmagoj Saric

> 1.2 Adopt agenda
>
Yes

> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
> All C++ reflector are now moved to listserv
>
> https://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
C++20 progress

>
> 2.2 Paper reviews
>
> Review Belfast results.
>
> 2.2.1: ML topics
>
> Richard Dosselman
>
> P1708R1: Math proposal for Machine Learning
>
>
https://docs.google.com/document/d/1VAgcyvL1riMdGz7tQIT9eTtSSfV3CoCEMWKk8GvVuFY/edit
>
Now in latex
for these simple stat functions, should they be stand alone (simpler to

implement as well if you want to add new functions, but now you have
multiple scan of data if you wnat to do all the algorithms), or aggregate
those into a class type structure (can do a single scan for multiple
algorithm similar to boost accumulate)
Vote results from belfast:
Should the author continue with the functionality and direction of
accumulator set or return to version 1?
(SF = accumulator_set, SA = version 1)

SF	F	N	A	SA
3	7	2	0	0

Favor accumulator set

But how do we add new functions and extend but that adds accessing the guts

Mode require passing a range and fill it up for all modes,
concern about havign multiple modes in the set of data,
Jens did not like returing a vector as it allocates memory, most STL pass
their own preallocated data structure

BFS and DFS traversal requires something allocated in the back, so no
allocation may be too hard a restriction
Can still extend for images or a set of images for facial recognition for
de-nosing
also Belfast asked for support for complex numbers

>
> P1709R1: Graph Proposal for Machine Learning
>
>
>
https://docs.google.com/document/d/1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE
/edit?usp=sharing
> <
>
https://nam02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.google.com%2Fdocument%2Fd%2F1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE%2Fedit%3Fusp%3Dsharing&data=02%7C01%7CPhil.Ratzloff%40sas.com%7C729b2cf8502641e4ae5e08d749064578%7Cb1c14d5c362545b3a4309552373a0c2f%7C0%7C0%7C637058163592253027&sdata=
4UQm8tqrcUbiZsr200UMrOaEModJYGNgP1oNot9PbAg%3D&reserved=0>
>
>
reviewing Belfast feedback
this algorithmic centric approach was good to continue, and in conjunction
with range based interpretation of graphs, and also doign it at compile

time with constexpr
and finally also address parallel executon, also question about using simd

ok constexpr was an oversight, so updated the read-only functions, unclear
whether we should change others
parallelism should be per- algorithm ? Yes seems reasonable, parallel graph
algo are not well understood, the best parallel graph algo is not just a
parallelized version of sequential, so I dont think that is a sensible algo
to have parallel DFS by just adding an execution policy, these things dont
parallelize, BFS also have synchronization issue; some have a priority
queue and requires single stepping

positive from Belfast
AL promotes a graph as range of ranges
BGL17 is a Better Graph Library using C++17
MS VS has range V3 and COncepts

So wanted to try it out with an example of adjacency array,
so look at how the graph is built, the ranges, and the implementation of
BFS and DFS
using the German route from the paper as starting point

vector of cities and edges
3 landas for cities and edge value first and seocnd

now we have pritning the structure
iterating through vertices, gettign key, outgoing edges, explciit for now
but could change to auto

BGL17 has bfs and dfs based on both nodes and vertexes

now using an iterator based on approach be cause it has depth
if not care about depth then use the range based which is second solution

the boolean is it always recorded? Yes always recorded
can we overrde the implementation? BGL17 philosophy was trying to get as
far away from original BGL
can put al lvisitor code in the body of the range based adaptor; I used BGL
and I thought that was useful
I was doign an experiment to see if this works to solve both flexibility
and performance as vsiitor can take a hit with performance

we could create a visitor that visits both edge and vertices, but it would
be more complcated to write; so MG if you can think of an example where
this does not work that would be great? I cant think of how such an eaxmple
would look like.

Centrality would be a great next algorithm,

the final one is BFS edge

now look at BFS/DFS implementation

One interestign observation is that range itself contains the content ot
state of the iteration, so as you are iteratign through so if you call
begin, it returns location of where you iterated to; this allows itrator to
be copied in a light weigh way

so need a const begin ...

created a concept called SearchableGrpahConcept
has a forward range,
to get begin and end of edges, we need to pass in the graph for context
this i causing us to have free function

so in bfs and dfs we dont have to worry whether it is directed, we just
wnat teh range of graphs to default to a set of vertices
but if you want begin of graph it returns the begin of the vertex collection
if you want begin of vertex it is begin of outedge

the ago is based on having an integral vertex key, so if we dont have an
integral vertex key, these algo wont work
yes, we wanted to have this key type that we can dereference n both the
inner and outer range
so maximal set only works on an edge list

with algorithmic centric approach, different ones will have different
requirements, we might be able to boil it down to a small list of
requirements

so enfornce to start an edge list, or user gives you an adjacency list
directly for the definition of the graph

in general, different algo knows what requirement they have, cocept define
formally what those algorithms are

we can also have nested concepts,

if it supports both the nwill have to do somethign extra to select. This
needs experiment in future

if we dont enable that design then we are lost, as we are trying to follow STL separation of algo and containers, he intent here is that we have free functions and we can create a single structure with the types

the whole properties might have issue when properties is mutable

data members are graph stack, vector bool
this boils down to how this is constructed

does vertex return the from or to? the to for directed graph
grabbing the key could be a hit for performance, if we know we have integers that have no need for hasing with key labels, why are we dealing with it
define concept for speed version
also about code size
OK so we have a template parameter in the definition of the graph? Right and if we check if it is consecutive, the nskip key lookup, as these can make a big difference

adjacency array with edges stored consecutively a vector or DAG, real impl is kept in impl file along with free functions that tie this to adjanceny array

can have this directed on any type or any structure

so we need to pass allocator since we cant add anything

cant prevent user from using the free functions, may be consequence so need education

dont look directly to see a graph,

no public funcions for removign edges so its semi-mutable

constructor have 2 template statements, one for the graph and the other for the constructor itself

static usage of grapph requires separet th container from algorithm so I can define a struct of my own that obeys an interface?
then we can statically allocate the value then we wont have to wait for constexpr STL

perhaps library can have a tool that generate static graph from dynamic ones, like serialize an unordered map translates a dynamic TF graph to a static C++ graph

XLA and glow will analyze a graph structure in any format like onyx nnef, and do fusioin transformation, reordering, and generate machine code , so I am lookign at this proposal to generate reusable tool for these graph

next iteration of paper is more examples, and more functions, and implementing the algo

suggest we also have an undirected graph and tha twould help. AL mentions other ones in BGL17 would also help

> Differentiable Programing by Marco Foco
>
Workign on it for next call

> P1416R1: SG19 - Linear Algebra for Data Science and Machine Learning
>
> https://docs.google.com/document/d/1IKUNiUhBgRURW-UkspK7fAAyIhfXuMxjk7xKikK4Yp8/edit#heading=h.tj9hitg7dbtr
>
> P1415: Machine Learning Layered list
>
>
https://docs.google.com/document/d/1elNFdIXWoetbxjO1OKol_Wj8fyi4Z4hogfj5tLVSj64/edit#heading=h.tj9hitg7dbtr
>
> 2.2.2 SG14 Linear Algebra progress:
> Different layers of proposal
>
>
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit
>
> 2.2.3 any other proposal for reviews?
>
> 2.3 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
>
> -2020-02-10 to 15: Prague, Czech Republic
>
> - 2020-06-01 to 06: Bulgaria
> - 2020-11-6-14: (New York, tentative)

> - 2021-02-22 to 27: Kona, HI, USA
>
> 3. Any other business
>
> New reflector
>
> http://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> Old Reflector
> https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19
> <https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg14>
>
> Code and proposal Staging area
>
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda
>
> TBD
>
> 5.2 Future meeting
>
> Dec 12, 2019 01:00 PM
> Jan 9, 2020 01:00 PM: Jan 13 is mailing deadline
> Feb 13, 2020 01:00 PM
> Mar 12, 2020 01:00 PM

Minutes for 2020/01/09 SG19 Conference Call

> 1.1 Roll call of participants
>
Phil Ratsloff, Marco Foco, Andrew Lumsdaine, jesun Firoz, Kdeweese,
Michael Wong, Matthew Galati, Vaibhav, william tambellini, Jens Maurer

> 1.2 Adopt agenda
>
> 1.3 Approve minutes from previous meeting, and approve publishing
> previously approved minutes to ISOCPP.org
>
> 1.4 Action items from previous meetings
>
> 2. Main issues (125 min)
>
> 2.1 General logistics
> All C++ reflector are now moved to listserv
>
> https://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
Lists of features we are working on:
https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0

>
> 2.2 Paper reviews
>
> Review Belfast results.
>
> 2.2.1: ML topics
>
> Richard Dosselman
>
> P1708R1: Math proposal for Machine Learning
>
>
https://docs.google.com/document/d/1VAgcyvL1riMdGz7tQIT9eTtSSfV3CoCEMWKk8GvVuFY/edit
>
> P1709R1: Graph Proposal for Machine Learning
>

R2 proposal
https://docs.google.com/spreadsheets/d/1JnUJBO72QVURttkKr7gn0_WjP--P0vAne8JBfzbRiy0/edit#gid=0
have 2 graph data structures
1 for directed and 1 for ordered graph
last telecon we show DFS and BFS implemented as ranges

what we have done
also have undirected graphs - what is the difference between directed and undirected?
- both have vertex and edges, but undirected edges are edges, for directed there are out and in edges, was trying to use a uniform api to remove that distinction
- still one place we use in and out where edge refers to 2 vertices
implemented shortest paths,
also have transitive closure
I have added category types, on whether we have sparse or dense graph,
a directed graph is a uniform graph and an out directed graph
bidirected, and unidirected follows
a sparse graph and a vertex range
request definition of dense (a matrix, random access inner, forward iterator outter) vs sparse graph (random access outter, forward iterator to inner)
we already have existing algo in library for textsearches
propose remove sparse and dense graph as it is implementation details
why integral vertex key? internal implementation requires we have integer key to look up vertex, vector in this case, want to relax overtime
the less restrictive one is random access iterator, so a DAG has random access iterators
what is teh key? -a DAG the key is from the vertex of container, for a map the key is defined by user
should contrain on OutIter result_iter please
need to specify restrictions on G and so use concepts, same with OutIter

shortest path has arbitrary list so has to do memory allocation
shortest distance has no memory allocation
BGL17 approach didnt really have shortest path or BFS do anything other then go through the graph
when u get to last vertex for the path, want to iterate each vertexes of the path, so have inner and outer loop you need to walk through

will you output distance to targets with a filter? No there is no filter
shortest distance and shortest paths give you the same opotion with leaves_only, so if you want mre sophisticated filters, it is possible
Do you really need leaves_only as a parameter; shortest_paths would only need the parent, any partial path is also the shortest path, for Dijkstra,

just store the parent pointer instead of constructing the paths
so give them a range of what we have now, with the distance may be a better way of doing that
another output u may want is an edge iterator, using a pair , along with a vertex iterator

parent and distance, may want a path of vertexes directly; instead of returning a vector of things, we just return a range
ok I will go back and work on getting both, implementation later after the mailing deadline

what examples of graph data structure should we have?
compressed sparse row (CSR) is useful, for the user to build their own
concern that we are interpreting graph as ranges of ranges,notion of dereferecing can be cumbersome, - that depends on on what algorithm actually needs, so a compressed data structure is useful to meet the need of the algorithm, so you can use MKL under the hood, may need to sync with Intel, graph-BLAS as an extension of MKL, using CSR optimizing for AVX
sparse LA or graphs are memory bound, theer is not any optimization you can apply to move things in memory any faster
CSR and adjacency edge list discussion
approach Intel Moscow team to see how they want to participate

in our case, we are not distinguishign dense vs sparse graph based on Jens comment

adapting this to their own data structues, - yes Jens mention wantign interface to be minimal ,if std library has equivalent things, then we shoudl reuse them.
I will look at CSR options

R1 proposal

>
>
https://docs.google.com/document/d/1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE/edit?usp=sharing
> <
>
>
https://nam02.safelinks.protection.outlook.com/?url=https%3A%2F%2Fdocs.google.com%2Fdocument%2Fd%2F1QkfDzGyfNQKs86y053M0YHOLP6frzhTJqzg1Ug_vkkE%2Fedit%3Fusp%3Dsharing&data=02%7C01%7CPhil.Ratzloff%40sas.com%7C729b2cf8502641e4ae5e08d749064578%7Cb1c14d5c362545b3a4309552373a0c2f%7C0%7C0%7C637058163592253027&sdata=4UQm8tqrcUbiZsr200UMrOaEModJYGNgP1oNot9PbAg%3D&reserved=0>
>

>
> P1707R0: Differentiable Programming by Marco Foco
>
explain differences automatic, vs symbolic differentiation
- auto works on algorithm ,and uses teh structure of the algorithm
- sym only works on expressions
also motivate why we want langage solution and not a library solution
-performance example demonstrate benefits
-but mainly you cannot do many things with library, need to use a specially named type as in Enoki
want to enable this auto diff tool kit for other things like matrices
matrix and vector types support is very important
assume people using this library is different then the library writer - so we will often combine multiple libraries together
https://eigen.tuxfamily.org/dox/unsupported/group__AutoDiff__Module.html
prompt Andrew L on Tulsa library
example: https://joelcfd.com/automatic-differentiation/

a fork of Eigen "aimed at Algorithmic Differentiation (AD) ":
https://gitlab.stce.rwth-aachen.de/stce/eigen-ad
A recent paper:
https://arxiv.org/abs/1911.12604

Numerical Differentiation - a new paper
Could be a library solution which can land earlier
limit and choosing h
sqrt in C++20 is still not constexpr
is there a bets practice for moving instead of copying a function when it is a pure function: let lewg decide
e.g. matrix solver ,so good to memoize f(x), and repeatedly call it with different h
AL to send reference
complex step is not really a numerical solution

> P1416R1: SG19 - Linear Algebra for Data Science and Machine Learning
>
> https://docs.google.com/document/d/1IKUNiUhBgRURW-UkspK7fAAyIhfXuMxjk7xKikK4Yp8/edit#heading=h.tj9hitg7dbtr
>
> P1415: Machine Learning Layered list
>
>
https://docs.google.com/document/d/1elNFdIXWoetbxjO1OKol_Wj8fyi4Z4hogfj5tLVSj64/edit#heading=h.tj9hitg7dbtr
>
> 2.2.2 SG14 Linear Algebra progress:

> Different layers of proposal
>
>
https://docs.google.com/document/d/1poXfr7mUPovJC9ZQ5SDVM_1Nb6oYAXlK_d0ljdUAtSQ/edit
>
> 2.2.3 any other proposal for reviews?
>
> 2.3 Other Papers and proposals
>
> 2.5 Future F2F meetings:
>
Paper submission:
https://isocpp.org/papers

> 2.6 future C++ Standard meetings:
> https://isocpp.org/std/meetings-and-participation/upcoming-meetings
>
> -2020-02-10 to 15: Prague, Czech Republic
>
> - 2020-06-01 to 06: Bulgaria
> - 2020-11: (New York, tentative)
> - 2021-02-22 to 27: Kona, HI, USA
>
> 3. Any other business
>
> New reflector
>
> http://lists.isocpp.org/mailman/listinfo.cgi/sg19
>
> Old Reflector
> https://groups.google.com/a/isocpp.org/forum/#!newtopic/sg19
> <https://groups.google.com/a/isocpp.org/forum/?fromgroups=#!forum/sg14>
>
> Code and proposal Staging area
>
> 4. Review
>
> 4.1 Review and approve resolutions and issues [e.g., changes to SG's
> working draft]
>
> 4.2 Review action items (5 min)
>
> 5. Closing process
>
> 5.1 Establish next agenda

>
> TBD
>
> 5.2 Future meeting
>
> Dec 12, 2019 01:00 PM
> Jan 9, 2020 01:00 PM: Jan 12 is mailing deadline
> Feb 13, 2020 01:00 PM
> Mar 12, 2020 01:00 PM