HyperFORM – Hyperlogs with FORM

Adam Kardos

in collaboration with

Sven Moch and Oliver Schnetz

Liverpool, UK, 2025, June 13th

UNIVERSITY of DEBRECEN

Contents

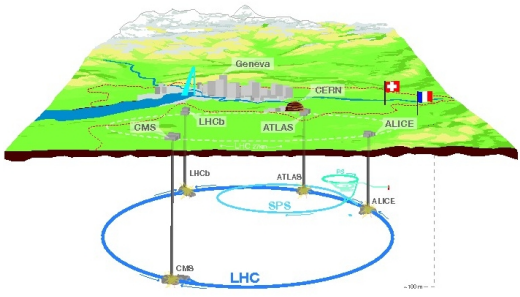
- Introduction
- Hyperlogs
- Hyperlogs for loop integrals
- Gentle look at fibration basis
- Unit Driven Development (UDD)
- UDD in `FORM`
- `HyperFORM`
- Summary and Outlook

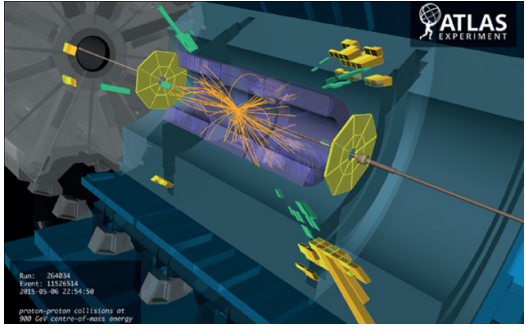
UNIVERSITY of DEBRECEN

Introduction

Introduction

LHC puts out very precise results (uncertainty less than 1 % in some cases)!

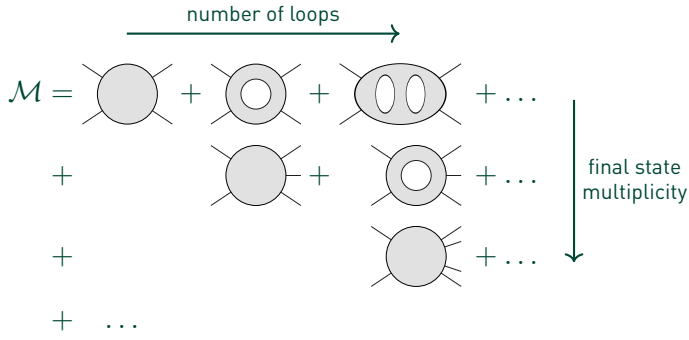




UNIVERSITY of DEBRECEN

Introduction

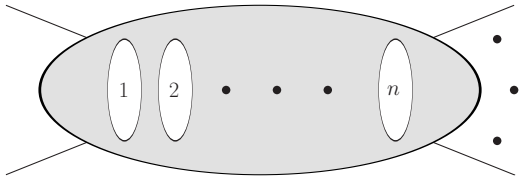
- Precision phenomenology is impossible without loop corrections
- Perturbative expansion of matrix element naturally involves loop corrections:



UNIVERSITY of DEBRECEN

Introduction

- Several methods to reduce loop integrals to a basis (to masters)
- Several ways to evaluate remaining (master) loop integrals
- For a class of integrals Feynman parameter representation and hyperlogs are working (well)



UNIVERSITY of DEBRECEN

Emergence of hyperlogs

Emergence of hyperlogs

Feynman parametrization of loop integrals:

$$I = \frac{e^{L\epsilon\gamma_E}\Gamma\left(\nu - \frac{Ld}{2}\right)}{\prod_{i=1}^{N}\Gamma(\nu_i)} \left(\prod_{i=1}^{N}\int_0^\infty dx_i\right) \delta\left(1 - \sum_{i=1}^{N} x_i\right) \left(\prod_{i=1}^{N} x_i^{\nu_i - 1}\right) \frac{\mathcal{U}^{\nu - \frac{L+1}{2}d}}{\mathcal{F}^{\nu - \frac{Ld}{2}}},$$

$$\nu = \sum_{i=1}^{N} \nu_i\,, \quad d = 4 - 2\epsilon$$

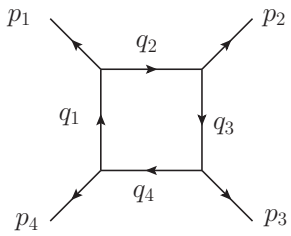
$\mathcal{U}$ and $\mathcal{F}$ are graph polynomials, $\mathcal{U}$ related to spanning trees of the graph, while $\mathcal{F}$ related to spanning 2-forests

For details see e.g.: Weinzierl: Feynman Integrals, chapter 3

UNIVERSITY of DEBRECEN

Hyperlogs – parametric representation – examples

- Simple one-loop massless box diagram:



$$
\begin{aligned}
q_1 &= \ell\,, \\
q_2 &= \ell - p_1\,, \\
q_3 &= \ell - p_1 - p_2\,, \\
q_4 &= \ell - p_1 - p_2 - p_3
\end{aligned}
$$

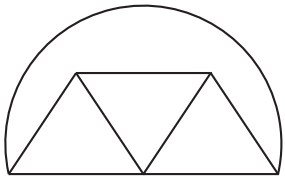
ℓ: loop momentum

$$
\begin{aligned}
\mathcal{U} &= x_1 + x_2 + x_3 + x_4\,, \\
\mathcal{F} &= (p_1 + p_4)^2 x_1 x_3 + (p_1 + p_2)^2 x_2 x_4
\end{aligned}
$$

UNIVERSITY of DEBRECEN

Hyperlogs – parametric representation – examples

- Zig-zag 4:



$$\begin{aligned}\mathcal{U} = {} & x_4x_6x_7x_8 + x_2x_6x_7x_8 + x_1x_6x_7x_8 + x_3x_4x_7x_8 + \\ & + x_1x_4x_7x_8 + x_2x_3x_7x_8 + x_1x_3x_7x_8 + x_1x_2x_7x_8 + \\ & + x_4x_5x_6x_8 + x_2x_5x_6x_8 + x_1x_5x_6x_8 + x_2x_4x_6x_8 + \\ & + x_1x_4x_6x_8 + x_3x_4x_5x_8 + x_1x_4x_5x_8 + x_2x_3x_5x_8 + \\ & + x_1x_3x_5x_8 + x_1x_2x_5x_8 + x_2x_3x_4x_8 + x_1x_3x_4x_8 + \\ & + x_1x_2x_4x_8 + x_4x_5x_6x_7 + x_2x_5x_6x_7 + x_1x_5x_6x_7 + \\ & + x_3x_4x_6x_7 + x_1x_4x_6x_7 + x_2x_3x_6x_7 + x_1x_3x_6x_7 + \\ & + x_1x_2x_6x_7 + x_3x_4x_5x_7 + x_1x_4x_5x_7 + x_2x_3x_5x_7 + \\ & + x_1x_3x_5x_7 + x_1x_2x_5x_7 + x_3x_4x_5x_6 + x_2x_4x_5x_6 + \\ & + x_2x_3x_5x_6 + x_1x_3x_5x_6 + x_1x_2x_5x_6 + x_2x_3x_4x_6 + \\ & + x_1x_3x_4x_6 + x_1x_2x_4x_6 + x_2x_3x_4x_5 + x_1x_3x_4x_5 + \\ & + x_1x_2x_4x_5 \\ \mathcal{F} = {} & 1\end{aligned}$$

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

Observations:

- The first graph (Symanzik) polynomial ($\mathcal{U}$) is always linear in x's
- The second graph (Symanzik) polynomial ($\mathcal{F}$) is only linear in absence of internal masses

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

- Assume the parametric representation is ϵ finite or a priori ϵ regularized:

$$\mathcal{U}^{\nu-\frac{\mathsf{L}+1}{2}\mathsf{d}} = \mathcal{U}^{\nu-2(\mathsf{L}+1)}\left[1+\epsilon(\mathsf{L}+1)\log\mathcal{U}+\mathcal{O}(\epsilon^2)\right],$$
$$\mathcal{F}^{\frac{\mathsf{L}\mathsf{d}}{2}-\nu} = \mathcal{F}^{2\mathsf{L}-\nu}\left[1-\epsilon\mathsf{L}\log\mathcal{F}+\mathcal{O}(\epsilon^2)\right], \quad \nu\in\mathbb{N}$$

- Here we stick to the lowest order term in ϵ

Emergence of hyperlogs

- Keep graph polynomial exponents formal:

$$\left(\prod_{i=1}^{N}\int_0^\infty \mathrm{d}x_i\right)\delta\left(1-\sum_{i=1}^{N}x_i\right)\frac{1}{\mathcal{U}^m\mathcal{F}^n}$$

- Using the Chen-Wu theorem we can keep a subset of integration variables in the Dirac-delta:

$$\left(\prod_{i=1}^{N}\int_0^\infty \mathrm{d}x_i\right)\delta\left(1-x_N\right)\frac{1}{\mathcal{U}^m\mathcal{F}^n}=\int_0^\infty \mathrm{d}x_1\cdots x_{N-1}\,\frac{1}{\mathcal{U}^m\mathcal{F}^n}\bigg|_{x_N=1}\,,\quad m,\,n\geq 1$$

$$\widetilde{\mathcal{U}}=\mathcal{U}|_{x_N=1}\,,\quad \widetilde{\mathcal{F}}=\mathcal{F}|_{x_N=1}$$

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

- Have to fix a σ integration order

$$\mathsf{x}_{\sigma(1)},\ \mathsf{x}_{\sigma(2)},\ldots,\mathsf{x}_{\sigma(\mathsf{N}-1)}$$

- Without internal masses both $\mathcal{F}$ and $\mathcal{G}$ are linear in all remaining xs

$$\int_0^\infty \mathrm{d}\mathsf{x}_1\cdots\mathsf{x}_{\mathsf{N}-1}\frac{1}{\widetilde{\mathcal{U}}^{\mathsf{m}}\widetilde{\mathcal{F}}^{\mathsf{n}}}=\int_0^\infty \mathrm{d}\mathsf{x}_1\cdots\mathsf{x}_{\mathsf{N}-1}\frac{1}{\left(\widetilde{\mathcal{U}}_1+\widetilde{\mathcal{U}}^{(1)}\mathsf{x}_{\sigma(1)}\right)^{\mathsf{m}}\left(\widetilde{\mathcal{F}}_1+\widetilde{\mathcal{F}}^{(1)}\mathsf{x}_{\sigma(1)}\right)^{\mathsf{n}}}$$

- Integrand can be partial fractioned in $\mathsf{x}_{\sigma(1)}$
- Resulting integrand has polynomial in denominator with some k power

$$\left(\mathsf{G}_1+\mathsf{G}^{(1)}\mathsf{x}_{\sigma(1)}\right)^{-\mathsf{k}}\begin{cases}\mathsf{k}\geq 2 & \text{rational function}\\ \mathsf{k}=1 & \text{logarithm}\end{cases},\qquad \mathsf{G}\in\{\widetilde{\mathcal{U}},\widetilde{\mathcal{F}}\}$$

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

Can we do something better than logarithms?
Hyperlogarithms!

- Special class of iterated integrals (Chen, The Annals of Mathematics 97 (2) (1973) 217246)

$$\mathrm{L}_{\omega_\sigma \mathrm{w}}(z) = \int_0^z \frac{\mathrm{d}z'}{z' - \sigma} \mathrm{L}_{\mathrm{w}}(z')\,, \quad \mathrm{L}_{\emptyset}(z) = 1$$

- Fulfill shuffle relations:

$$\mathrm{L}_{\mathrm{w}_1}(z)\mathrm{L}_{\mathrm{w}_2}(z) = \mathrm{L}_{\mathrm{w}_1}(z) \sqcup\!\sqcup \mathrm{L}_{\mathrm{w}_2}(z)$$

- See also the slides of Coenraad from yesterday

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

Hyperlogarithms:

- Special, one-dimensional Chen iterated path integrals
  Chen, Transactions of the American Mathematical Society, Vol. 156, 359-379 (1971)
- These also called Goncharov polylogarithms,
  A. B. Goncharov, Math. Research Letters. 5(4), 497 (1998), A. Goncharov, Multiple polylogarithms and mixed Tate motives (2001)
- Can get harmonic polylogarithms if letters from $\{-1, 0, 1\}$
  E. Remiddi, J. A. M. Vermaseren, Int.J.Mod.Phys. A15 (2000) 725–754
- As special cases we can get classical polylogarithms and multiple polylogarithms

UNIVERSITY of DEBRECEN

Emergence of hyperlogs

Hyperlogarithms:

- Most important developments were done by Francis Brown
  - Annales scientifiques de l'Ecole Normale Superieure 42 (3) (2009) 371–489. arXiv:math/0606419
  - Communications in Mathematical Physics 287 (2009) 925–958. arXiv:0804.1660, doi:10.1007/s00220-009-0740-5
  - On the periods of some Feynman integrals, ArXiv e-print: arXiv:0910.0114
- Our treatment of loop integrals with hyperlogs closely follow:
  E. Panzer, Computer Physics Communications, 188, 148–166 (2015)

What could be done with these constructs?

UNIVERSITY of DEBRECEN

- With this definition first integration becomes:

$$\int_0^\infty dx_{\sigma(1)} \int_0^\infty dx_{\sigma(2)} \cdots \int_0^\infty dx_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G_1 + G^{(1)} x_{\sigma(1)}} =$$

$$= \int_0^\infty dx_{\sigma(1)} \int_0^\infty dx_{\sigma(2)} \cdots \int_0^\infty dx_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G^{(1)}} \frac{1}{x_{\sigma(1)} + G_1/G^{(1)}}$$

$$= \int_0^\infty dx_{\sigma(2)} \cdots \int_0^\infty dx_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G^{(1)}} L_{\omega_{-G_1/G^{(1)}}}(\infty)$$

- This form is not without problems...

$$\int_0^\infty \mathrm{d}\mathsf{x}_{\sigma(2)} \cdots \int_0^\infty \mathrm{d}\mathsf{x}_{\sigma(\mathsf{N}-1)} \frac{\mathcal{R}(\mathsf{x}_{\sigma(2)}, \ldots, \mathsf{x}_{\sigma(\mathsf{N}-1)})}{\mathsf{G}^{(1)}} \mathsf{L}_{\omega_{-\mathsf{G}_1/\mathsf{G}^{(1)}}}(\infty)$$

$$\int_0^\infty \mathrm{d}x_{\sigma(2)} \cdots \int_0^\infty \mathrm{d}x_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G^{(1)}} L_{\omega}{}_{-G_1/G^{(1)}}(\infty)$$

- Only the full Feynman integral is finite individual terms can diverge

UNIVERSITY of DEBRECEN

$$\int_0^{\infty} \mathrm{d}x_{\sigma(2)} \cdots \int_0^{\infty} \mathrm{d}x_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G^{(1)}} L_{\omega_{-G_1/G^{(1)}}}(\infty)$$

- Only the full Feynman integral is finite individual terms can diverge
- All upper integration limits are infinity (how to set up the iterated integrations?)

UNIVERSITY of DEBRECEN

$$\int_0^\infty \mathrm{d}x_{\sigma(2)} \cdots \int_0^\infty \mathrm{d}x_{\sigma(N-1)} \frac{\mathcal{R}(x_{\sigma(2)}, \ldots, x_{\sigma(N-1)})}{G^{(1)}} L_{\omega_{-G_1/G^{(1)}}}(\infty)$$

- Only the full Feynman integral is finite individual terms can diverge
- All upper integration limits are infinity (how to set up the iterated integrations?)
- Next integration variable also in letter: iterated form is violated

UNIVERSITY of DEBRECEN

Hyperlogs for loop integrals

Hyperlogs for loop integrals

Problem: Only the full Feynman integral is finite individual terms can diverge

- Introduce regularized limits
- Partially change the end-point

$$\mathsf{L}_{\mathsf{w}}(\infty) = \sum_{\mathsf{w}_1,\mathsf{w}_2} \mathsf{L}_{\Phi_{1/\mathsf{z}}(\mathsf{w}_1)}(0) \cdot \mathsf{L}_{\mathrm{reg}_0^\infty(\mathsf{w}_2)}(\infty)\,, \quad \Phi_{\mathsf{f}}(\omega_\sigma) = \omega_{\mathsf{f}(\sigma)} - \omega_{\mathsf{f}(\infty)}\,, \quad \omega_\infty = \emptyset$$

- Limits at infinity get replaced by regularized limits at infinity and/or replaced by limits at zero
- Limits at zero can be (depending on word [w] and prefactors):
  - expanded as a power series
  - logarithmic singularities can be identified which should cancel in the final result

UNIVERSITY of DEBRECEN

Hyperlogs for loop integrals

Problem: All upper integration limits are infinity (how to set up the iterated integrations?)
For an iterated integral we need:

$$\cdots \int_0^{x_{i+1}} dx_i \int_0^{x_i} \frac{dx_{i-1}}{x_{i-1}-\sigma} \sum_{w'} L_{w'}(x_{i-1}) \quad : \quad x_{i-1} \not\in w'$$

- To set up iterated integrals we rely on the next solution

Hyperlogs for loop integrals

Problem: Next integration variable also in letter: iterated form is violated

$$L_{\omega_{-G_1/G^{(1)}}}(\infty) = \sum_{w'} \lambda^{(w')} L_{w'}(x_{\sigma(2)})$$

- Fibration basis is needed to turn hyperlogs at infinity inside-out to reveal dependence on next integration variable
  - Most complicated algorithm to implement
  - Shuffle-regularization at zero
  - Log-differentiation of letters and their differences
  - Zero-limit in next integration variable avoiding singularities
  - Rescaling of word in $x_{\sigma(2)}$

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

Gentle look at fibration basis

- Important to continue the integration with next variable (t):

$$\lim_{z\to\infty} L_{w(t)}(z) = \sum_{w'} \lambda^{(w')} L_{w'}(t)$$

- It is more important to be able to carry this out for reguralized limit hyperlogs:

$$\lim_{z\to\infty} L_{\mathrm{reg}_0^\infty(w(t))}(z) = \sum_{w'} \lambda^{(w')} L_{w'}(t)$$

- For regularized limit hyperlog we use Panzer's notation as well:

$$\lim_{z\to\infty} L_{\mathrm{reg}_0^\infty(w(t))}(z) = \operatorname*{Reg}_{z\to\infty} L_w(z)$$

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

- We use the Panzer notation when words are explicit:

$$\mathsf{w} = \omega_{\sigma_1}\omega_{\sigma_2}\cdots\omega_{\sigma_n}$$

- The differentiation of a limit hyperlog w.r.t. variable t:

$$\partial_t \operatorname*{Reg}_{z\to\infty} \mathsf{L}_w(z) = -[\partial_t \log \sigma_n] \operatorname*{Reg}_{z\to\infty} \mathsf{L}_{\omega_{\sigma_1}\cdots\phi_{\sigma_n}}(z) + \\ + \sum_{i=1}^{n-1} [\partial_t \log(\sigma_i - \sigma_{i+1})] \cdot \left(\operatorname*{Reg}_{z\to\infty} \mathsf{L}_{\omega_1\cdots\phi_{\sigma_{i+1}}\cdots\omega_n}(z) - \operatorname*{Reg}_{z\to\infty} \mathsf{L}_{\omega_1\cdots\phi_{\sigma_i}\cdots\omega_n}(z) \right)$$

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

- The differentiation of a limit hyperlog w.r.t. variable t:

$$\partial_t \operatorname*{Reg}_{z\to\infty} L_w(z) = -[\partial_t \log \sigma_n] \operatorname*{Reg}_{z\to\infty} L_{\omega_{\sigma_1}\dots\psi_{\sigma_n}}(z) + \\ + \sum_{i=1}^{n-1} [\partial_t \log(\sigma_i - \sigma_{i+1})] \cdot \left(\operatorname*{Reg}_{z\to\infty} L_{\omega_1\cdots\psi_{\sigma_{i+1}}\cdots\omega_n}(z) - \operatorname*{Reg}_{z\to\infty} L_{\omega_1\cdots\psi_{\sigma_i}\cdots\omega_n}(z) \right)$$

  - Differential form has limit hyperlogs with words having one less letter
  - t dependence is partially exposed (reduced words can still have dependence on t)

  $\Rightarrow$ Serves as an iterative way to construct fibration basis

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

- Differential form is given to define fibration basis in variable t

$\Rightarrow$ Integration constant has to be provided:

$$\operatorname*{Reg}_{z\to\infty} L_{w(t)}(z) = \mathcal{C} + \sum_{w'} \lambda^{(w')} L_{w'}(t)$$

- Integration constant is defined as a regular $t \to 0$ limit of the original limit hyperlog:

$$\mathcal{C} = \operatorname*{Reg}_{t\to 0} \operatorname*{Reg}_{z\to\infty} L_w(z)$$

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

- Integration constant is defined as a regular $t \to 0$ limit of the original limit hyperlog:

$$\mathcal{C} = \underset{t \to 0}{\mathrm{Reg}} \, \underset{z \to \infty}{\mathrm{Reg}} \, \mathsf{L}_{\mathsf{w}}(z)$$

- The $t \to 0$ limit is highly non-trivial:
  - If last letter has a constant term $t \to 0$ can be safely taken
  - If leading term in last letter depends on t rescaling and/or shuffle regularization is needed

UNIVERSITY of DEBRECEN

Gentle look at fibration basis

To implement an efficient integration method using hyperlogs several non-trivial algorithms are needed, including

- Nested data-structures to hold polynomial and rational function letters
- Letters are forming words
- Have to detect variable dependence in function arguments
- Shuffling of argument lists
- Eliminating arguments
- Rescaling of arguments
- Selective differentiation and integration of terms
- Wanted the code to be updatable for a long time

$\Rightarrow$ Most sophisticated software development is selected

UNIVERSITY *of* DEBRECEN

Unit Driven Development

UNIVERSITY of DEBRECEN

Unit Driven Development

Idea:

- Completely modular design
- Re-usable, as generic as possible routines
- Motto:

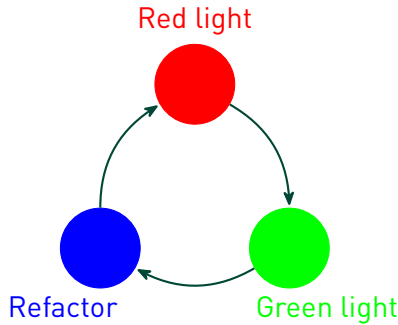
  If description of routine needs more than a single sentence it does too much
  $\Longrightarrow$ break it to smaller ones
- Algorithms are not etched to stone, need to be updatable

$\Rightarrow$ Functionality should be tested separately

$\Rightarrow$ Unit driven development philosophy is adopted

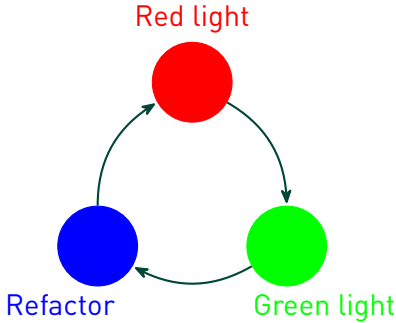
UNIVERSITY of DEBRECEN

Unit Driven Development



UNIVERSITY of DEBRECEN

Unit Driven Development

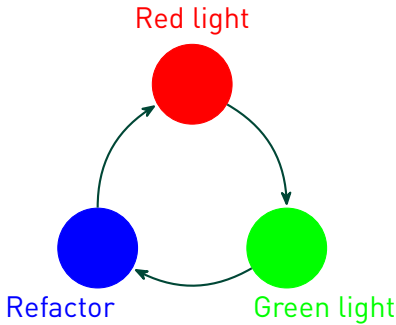


Red light:

- Start with an empty routine
- Create the unit test first
  - Specify an input
  - Define what output we would like to see
- With an empty routine the test will first fail "red light"

UNIVERSITY of DEBRECEN

Unit Driven Development



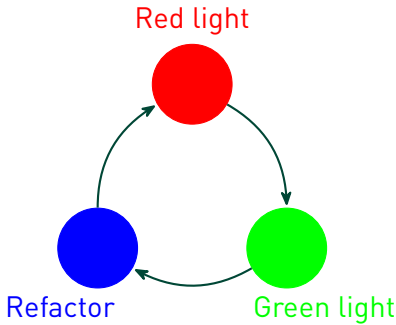
Green light:

- Implement a suitable algorithm
- Tweak it until the unit passes test

$\Rightarrow$ It does the desired functionality

UNIVERSITY of DEBRECEN

Unit Driven Development



Refactor:

- If performance is not satisfactory change the algorithm, improve the code
- It is safe to touch the routine because we will always see when we break functionality!

UNIVERSITY of DEBRECEN

UDD in FORM

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized
- Each routine receives its own unit test file

```
ArgsToRatFunc.frm                                DifferentiateHlogsAtLevel.frm                    linf-to-mzv.log
Babis.frm                                        DissectExprWithArgNumber.frm                     MergeLevelExpressions.frm
ChangeFuncHeadForGivenWordLengthAndVarDependence.frm  EncodeNumberofArgsInFunctionHead.frm        NonNumericArgsToExtraSymbols.frm
ChangeFuncHeadForSameArgFuncs.frm                ExtractFunctionsToExtraSymbols.frm               PartingForLimitDiffs.frm
ChangeFunctionHeadForVarDependentInstances.frm   ExtraSymbolsToExpression.frm                    RegularizeAndDifferentiateForFibrationBases.frm
ChangeHyperVarFull.frm                           FromHarmonicToMZV.frm                            RevealVarLog.frm
check.rb                                         FromLimitHlogsToNumbers.frm                      ShuffleRegularizeAtInfinity.frm
CollectHlogsForFibrationBasis.frm                FromLinfToLone.frm                               ShuffleRegularizeAtZero.frm
ConstructExpressionFromLargeTerms.frm            FromLoneToHarmonic.frm                           SimpleIntegrate.frm
ConvertDetailedRatFuncs.frm                      HyperExpand.frm                                  SimplifyDiffLogDiffFuncs.frm
CreateFunctionExpressionWithExtras.frm           IncludeTagInExprs.frm                            Simplify.frm
CreateRatFuncArgsInVar.frm                       IntegrateAtLevelSubstituteAbove.frm              SimplifyLimitHyperlogs.frm
DecomposeWRTvar.frm                              IntHlogDenZeroLetterRatFun.frm                   SimplifyRatFuncArguments.frm
DetermineMinAndMaxFunctionIndices.frm            IntHlogsV2.frm                                   SimplifyRatFuncsGCD.frm
DetermineUsedVarsInDollar.frm                    IntRatFun.frm                                    SimplifyRatFuncsNumeric.frm
DetermineUsedVarsInExpr.frm                      IsolateAndDissectVarDepFunctions.frm             SubstituteToExprs.frm
DetermineUsedVarsInLargeTerms.frm                IsolateAndDissectVarDepFunctionsThruExtra.frm    SubstituteToExprsWithExtrasAndTable.frm
DetermineUsedVarsInTerm.frm                      LimInfToLimZero.frm                              TagTerms.frm
DetermineZeroLimitsForFibrationBasis.frm         LimitsToHlogs.frm                                TakeZeroLimitInLimitHlog.frm
```

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized
- Each routine receives its own unit test file
- Create an empty routine

```
*{{{ FromLimitHlogsToNumbers
#procedure FromLimitHlogsToNumbers(ExprID,RegLinfFuncID,LinfFuncID)
  .sort:FromLimitHlogsToNumbers start;
  skip;
  nskip `ExprID';
*
*
  .sort:FromLimitHlogsToNumbers fin;
#endprocedure
*}}}
```

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized
- Each routine receives its own unit test file
- Create an empty routine
- Implement a bunch of tests

```
*{{{ FromLimitHlogsToNumbers_1 :
#include- declare-hyper.h
#include- hyper.h
#include- mzvlow.h
*
symbol a;
cfunction RegLinf;
cfunction Linf;
*
local F = RegLinf(-a);
*
.sort
#call FromLimitHlogsToNumbers(F,RegLinf,Linf)
print +s;
.end
assert succeeded?
assert result("F") =~ expr("
  + RegLinf(-a)
")
*}}}
```

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized
- Each routine receives its own unit test file
- Create an empty routine
- Implement a bunch of tests
- They will fail

```
================================================================================
Finished in 0.101172256 seconds.
--------------------------------------------------------------------------------
15 tests, 30 assertions, 12 failures, 0 errors, 0 pendings, 0 omissions, 0 not
ifications
20% passed
--------------------------------------------------------------------------------
148.26 tests/s, 296.52 assertions/s
```

UNIVERSITY of DEBRECEN

UDD in FORM

- Using `check.rb` provided with FORM slightly personalized
- Each routine receives its own unit test file
- Create an empty routine
- Implement a bunch of tests
- They will fail
- Implement functionality and test

```
adam@fedora:~/Documents/scientific/temp/hyper/check$ ./check.rb FromLimitHlogs
ToNumbers.frm
Check /home/adam/sciapps/bin/form
FORM 5.0.0-beta.1 (Mar  7 2025, v5.0.0-beta.1-122-g638f84b)  Run: Sun Jun  8 0
8:49:33 2025
Loaded suite ./check
Started
Finished in 0.108981223 seconds.
--------------------------------------------------------------------------------
15 tests, 30 assertions, 0 failures, 0 errors, 0 pendings, 0 omissions, 0 noti
fications
100% passed
--------------------------------------------------------------------------------
137.64 tests/s, 275.28 assertions/s
```

UNIVERSITY of DEBRECEN

HyperFORM

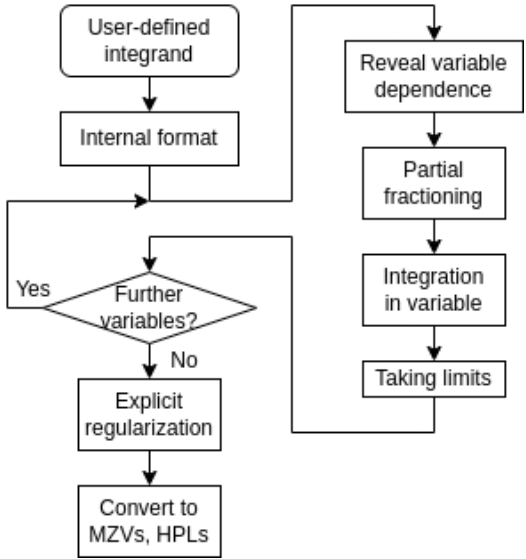
UNIVERSITY of DEBRECEN

HyperFORM

- All algorithms are implemented in FORM
- All major algorithms from
  E. Panzer, Computer Physics Communications, 188, 148–166 (2015)
- Can treat ϵ-finite integrals
- ϵ regularization is also possible, uses: E. Panzer, JHEP 03 (2014) 071
- Expansion of non-integer exponents
- Integration using the basis on rational functions and hyperlogs
- Series expansion of hyperlogs
- Taking limits of hyperlogs
- Conversion (when possible) to MZVs

UNIVERSITY of DEBRECEN

HyperFORM



UNIVERSITY of DEBRECEN

HyperFORM – Benchmarking

Use zig-zag diagrams for benchmarking:

- ϵ-finite
- Fibration basis generation is the key algorithm

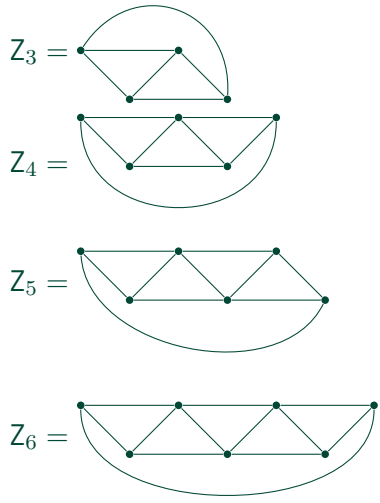
$\Rightarrow$ Finite multiloop diagrams are ideal to check performance

UNIVERSITY of DEBRECEN

HyperFORM – benchmark on zig-zags

Original implementation of E. Panzer, Computer Physics Communications, 188, 148–166 (2015) is in Maple as HyperInt

	HyperInt [s]	HyperFORM [s]
Z_3	0.8	0.05
Z_4	1.5	0.3
Z_5	54	17
Z_6	103000	29000



UNIVERSITY of DEBRECEN

Summary and Outlook

UNIVERSITY of DEBRECEN

Summary and Outlook

- Integration routines implemented in `FORM` using hyperlogs
- All key algorithms are present to be useful for Feynman integral calculations
- Generating fibration basis is highly non-trivial with many essential manipulations:
  - Shuffle regularization
  - Differentiation
  - Integration
  - Rescaling
  - Solving multiple equations
  - Back substitution
- Performance is convincing, there are a couple of ideas to implement...

UNIVERSITY *of* DEBRECEN

Thank you for your attention!