Project Caesar, a blueprint for numerical evaluation of Feynman integrals

based on:
arXiv:2201.02576
in collaboration with:

Ayres Freitas, Janusz Gluza, Krzysztof Grzanka, Martijn Hidding, Ievgen Dubovik

Johann Usovitsch

03. February 2022

Outline

Electroweak Precision Physics

	Experiment	**Theory uncertainty**	**Main source**
$M_W\,[\mathrm{MeV}]$	80385 ± 15	4	$N_f^2\alpha^3,\ N_f\alpha^2\alpha_\mathrm{s}$
$\sin^2\theta^\mathrm{l}_\mathrm{eff}\,[10^{-5}]$	23153 ± 16	4.5	$N_f^2\alpha^3,\ N_f\alpha^2\alpha_\mathrm{s}$
$\Gamma_Z\,[\mathrm{MeV}]$	2495.2 ± 2.3	0.4	$N_f^2\alpha^3,\ N_f\alpha^2\alpha_\mathrm{s},\ \alpha\alpha_\mathrm{s}^2$
$\sigma^0_\mathrm{had}\,[\mathrm{pb}]$	41540 ± 37	6	$N_f^2\alpha^3,\ N_f\alpha^2\alpha_\mathrm{s}$
$R_b = \Gamma^b_Z/\Gamma^\mathrm{had}_Z\,[10^{-5}]$	21629 ± 66	15	$N_f^2\alpha^3,\ N_f\alpha^2\alpha_\mathrm{s}$

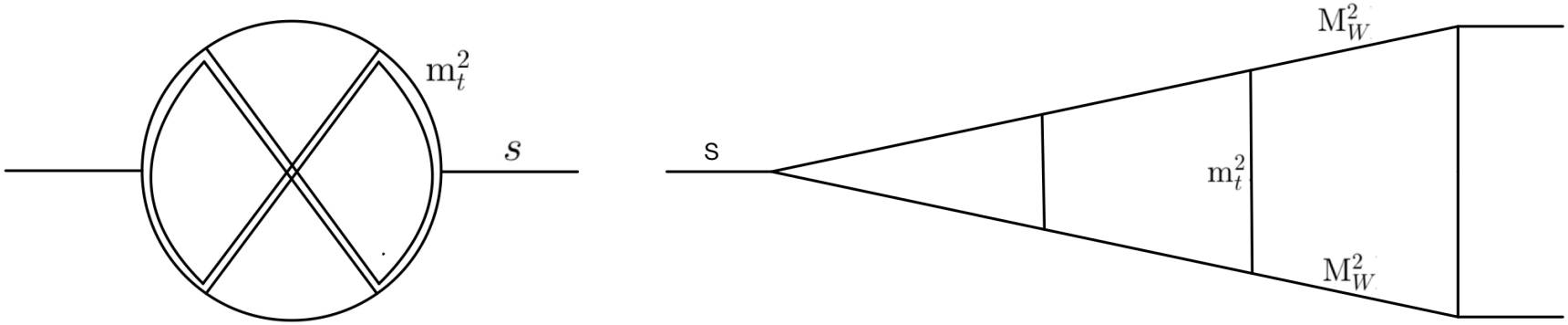
- The number of Z-bonos collected at LEP is 1.7×10^7
- Many pseudo observables are determined with high precision
- Present theoretical predictions (at least one order of magnitude better) are accurate enough to fullfill experimental demands

Overview Experiment Future

	Experiment uncertainty			**Theory uncertainty**	
	ILC	**CEPC**	**FCC-ee**	**Current**	**Future**
$M_W[\mathrm{MeV}]$	3-4	3	1	4	1
$\sin^2\theta^{\mathrm{l}}_{\mathrm{eff}}[10^{-5}]$	1	2.3	0.6	4.5	1.5
$\Gamma_Z[\mathrm{MeV}]$	0.8	0.5	0.1	0.4	0.2
$R_b[10^{-5}]$	14	17	6	15	7

- The concepts for the new experiments will have new demands to the theoreticle predictions
- The projection to the theory errors in the future assumes that the missing corrections $\alpha\alpha_{\mathrm{s}}^2$, $N_f^2\alpha^3$, $N_f\alpha^2\alpha_{\mathrm{s}}$ will become available
- Theoretical computations are universal

Samples of three-loop Feynman integrals



- We project all Feynman integrals to scalar integrals
- We need to compute all Feynman integrals only up to the finite order in $\epsilon = (4-d)/2$, d the space time dimension
- At the end we want to make sure we are able to compute all three-loop Feynman integrals appearing in e.g. the $Z\bar{b}b$ vertex numerically with as many significant digits as appropriate in physical kinematic regions, this statement is very fresh
- We perform the computations with automated tools

Grading the difficulty of a computation

- The integrals depend on up to four dimensionless parameters

$$\left\{ \frac{M_H^2}{M_Z^2}, \frac{M_W^2}{M_Z^2}, \frac{m_t^2}{M_Z^2}, \frac{(s+i\delta)}{M_Z^2} \right\} |_{s=M_Z^2} \tag{1}$$

- Many of them contain ultraviolet and infrared singularities, even though the divergences cancel in the final result
- Computations involve $\mathcal{O}(100)$ master integrals

Numerical Methods

Many formal successfull studies are available on the market.

- Loop tree duality [Capatti, Hirschi, Pelloni, Ruijl, 2021] progress of the automatisation is already in the implementation stage
- Unitarity cut techniques [Abreu, Ita, Page,Tschernow,2021] progress of the automatisation is already in the implementation stage
- pySecDec approach [Long Chen, Heinrich, Jones, Kerner, Klappert, Schlenk, 2021] fully automated
- Auxiliary mass flow [Brønnum-Hansen, Melnikov, Quarroz, Chen-YuWang, 2021], [Xiao Liu, Yan-Qing Ma, 2022] fully automated
- Solving a system of differential equations numerically [Lee, Smirnov, Smirnov, 2018], [Mandal, Zhao, 2019], [Moriello, 2019], [Bonciani, Del Duca, Frellesvig, Henn, Hidding, Maestri, Moriello, Salvatori, Smirnov, 2019], [Hidding, 2020], [Abreu, Ita, Moriello, Page, Tschernow, Zeng 2020] fully automated, publication is in accelerating progress

Feynman integral

$$T(a_1,\ldots,a_N) = \int \left(\prod_{i=1}^{L} \mathrm{d}^d \ell_i\right) \frac{1}{P_1^{a_1} P_2^{a_2} \cdots P_N^{a_N}}, \quad N = \frac{L}{2}(L+1) + LE \tag{2}$$

- $P_j = q_j^2 - m_j^2$, $j = 1,\ldots,N$, are the inverse propagators
- The momenta q_j are linear combinations of the loop momenta ℓ_i, $i = 1,\ldots,L$ for an L-loop integral, and external momenta p_k, $k = 1,\ldots,E$ for $E+1$ external legs
- The m_j are the propagator masses
- The a_j are the (integer) propagator powers
- Auxiliary mass flow [AMF] propagators are: $P_j = q_j^2 - m_j^2 + i\eta$
- AMF-ϵ matching is a brilliant tool which is on par with finite field methods in IBP reductions

Differential Equations

- Each family of Feynman integrals $T(a_1, \ldots, a_N)$ may be charcterized through a system of differential equations [Kotikov, 1991], [Remiddi, 1997], [Gehrmann, Remiddi, 2000]

$$\partial_{s_i} \vec{f} = M_{s_i}(s_i, \epsilon) \vec{f} \tag{3}$$

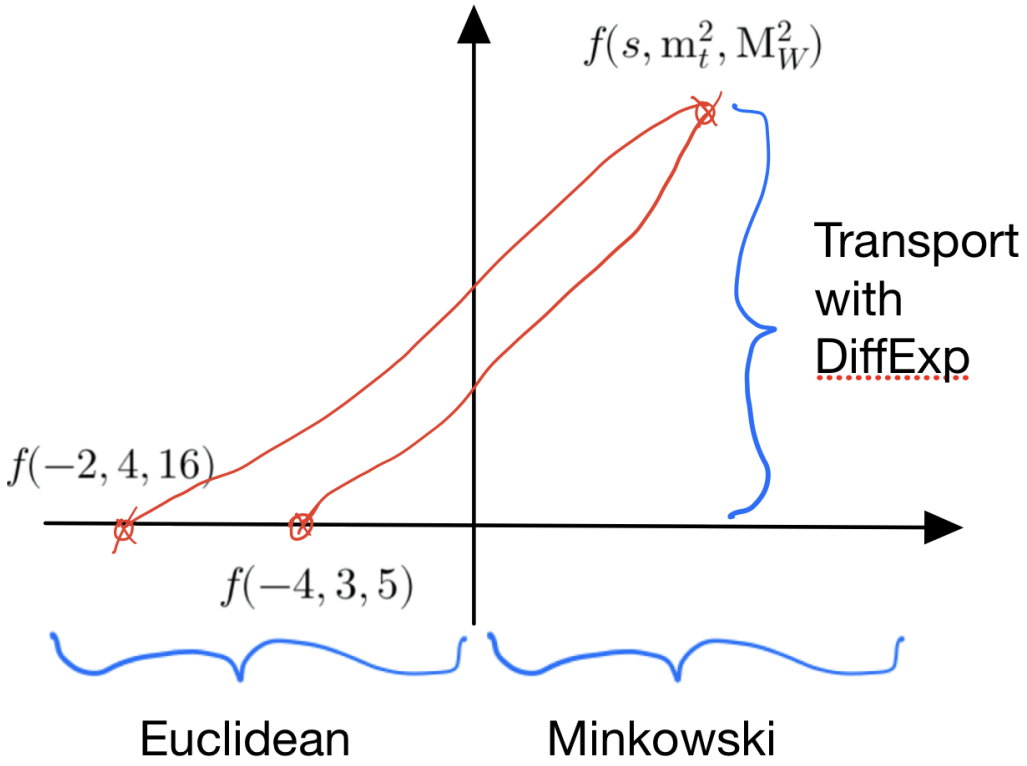
  and a set of master integrals $\vec{f}$
- We take derivatives in kinematic invariants and masses denoted as s_i in $\vec{f}$
- We express these derivatives again as a linear combination in terms of the same master integrals with the help of integration-by-parts identities [Chetyrkin, Tkachov, 1981]

The difficult part is to cast a physics problem in the form of Eq. (3). If this is done successfully, we have powerful tools to solve the physics problem.

Caesar: blueprint for numerical evaluation of Feynman integrals

- Developers team: Martijn Hidding and me.
- Basic idea: Caesar has an interface to Kira, Reduze 2 [Von Manteuffel, Studerus, 2012], (pySecDec [Borowka et al., 2018] or AMFlow [Xiao Liu, Yan-Qing Ma, 2022]) and DiffExp [Martijn Hidding, 2021].
- Kira - the backbone / major bottleneck of the Caesar project - solves linear system of equations
- Reduze 2 - finds candidates for a finite basis of master integrals
- pySecDec - computes these master integrals in Euclidean regions - boundary terms for the system of differential equations
- AMFlow - computes these master integrals in physical regions - boundary terms for the system of differential equations
- DiffExp - transports the boundary terms to an arbitrary physical point
- Error estimate: repeat the chain of tools for different initial boundary terms

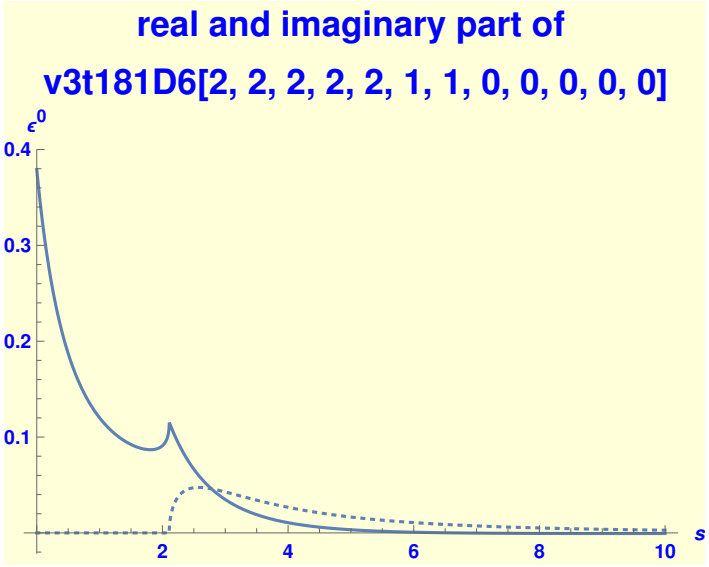
One Possible Application of Caesar



- All master integrals $f_i(...)$ are finite integrals (Reduze)
- Master integrals $f_i(...)$ are evaluated numerically in Euclidean regions (pySecDec) or with (AMFlow) $\rightarrow$ initial boundary terms
- System of differential equations is generated with (Kira)
- Use series expansion of the system of differential equations to transport from the initial boundary terms to Minkowskien physical regions (DiffExp)

The advantage of automated differential equations - ϵ^0

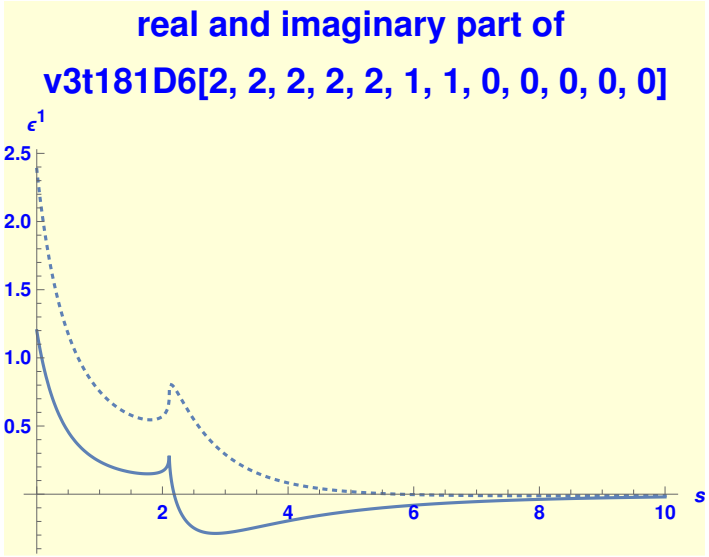
- While auxiliary mass flow gives numerical results for arbitrary feynman integrals one point at a time
- Automated differential equations give numerical results one line at a time.



- Timing: Total time: 254.769 sec, AMFlow one point 171 sec, DiffExp preparing the line: 12 sec
- Accuracy 20 digits

The advantage of automated differential equations - ϵ^1

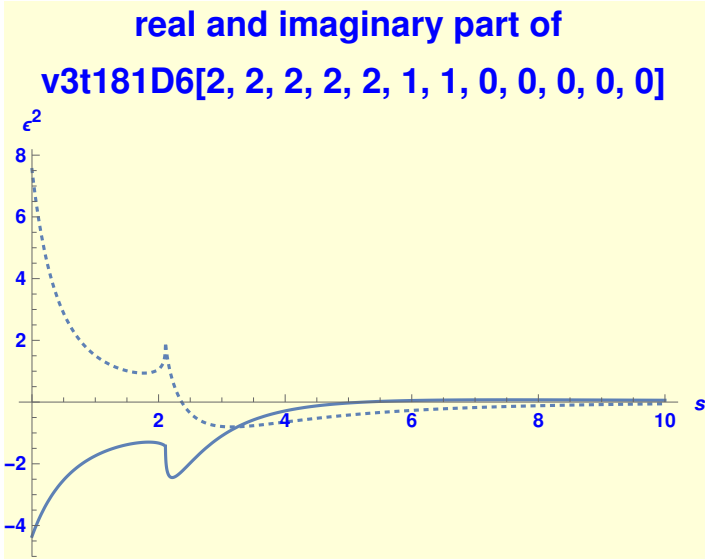
- While auxiliary mass flow gives numerical results for arbitrary feynman integrals one point at a time
- Automated differential equations give numerical results one line at a time.



- Timing: Total time: 254.769 sec, AMFlow one point 171 sec, DiffExp preparing the line: 12 sec
- Accuracy: 20 digits

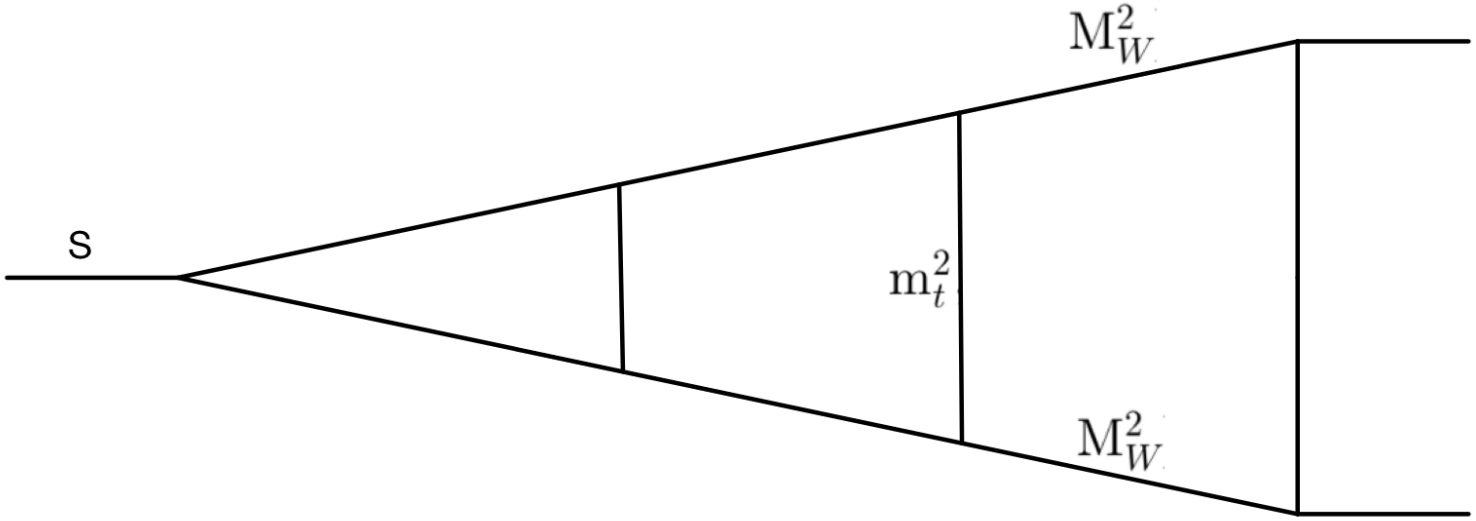
The advantage of automated differential equations - ϵ^2

- While auxiliary mass flow gives numerical results for arbitrary feynman integrals one point at a time
- Automated differential equations give numerical results one line at a time.



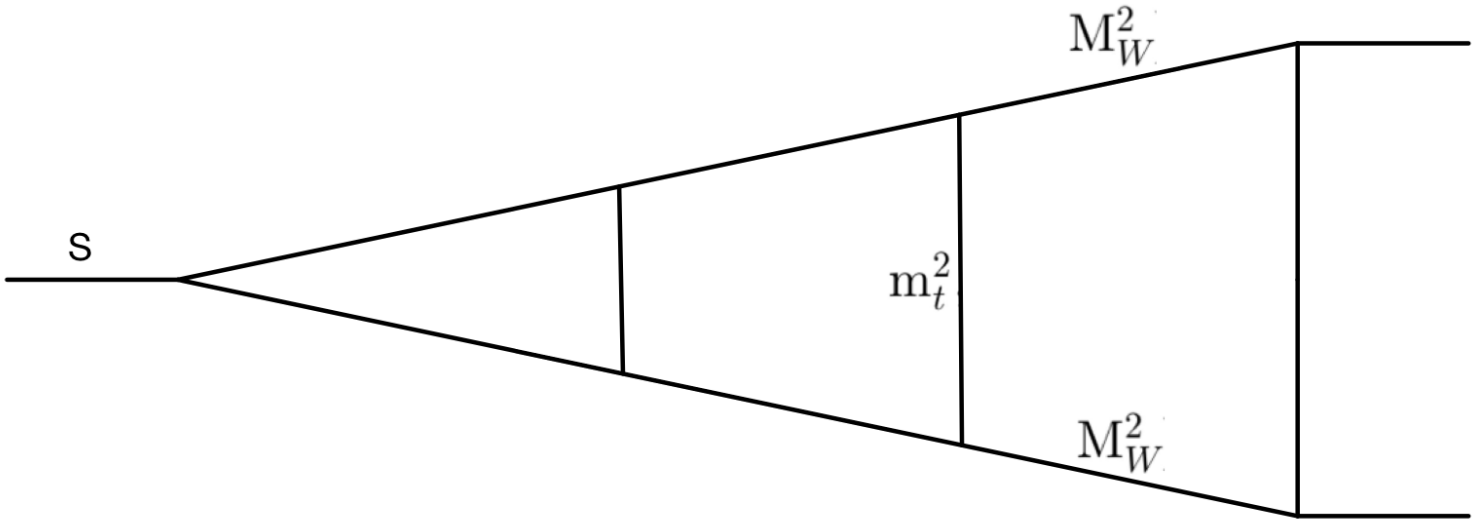
- Timing: Total time: 254.769 sec, AMFlow one point 171 sec, DiffExp preparing the line: 12 sec
- Accuracy 20 digits

Caesar: Integralfamily v3t181



- In Euclidean regions (s, M_W^2, m_t^2)=(-2,4,16)
  -> v3t181$^{d=4-2\epsilon}[1,1,1,1,0,1,1,1,1,0,0,0] =$
  $0.13395266644416018390274981 2$ with 25 significant digits
- At the present stage of the project this high accuracy was achieved semi-automatic
- Today AMFlow should give this precision fully automatically.
- 5 times longer run time compared to an 8 digit result (see next slide)

Caesar: Integralfamily v3t181



- In physical regions (s, M_W^2, m_t^2)=(1,$(\frac{401925}{4559382})^2$, $(\frac{433000}{227969})^2$)

  -> v3t181$^{d=4-2\epsilon}$[1, 1, 1, 1, 1, 1, 1, 1, 1, -3, 0, 0] =

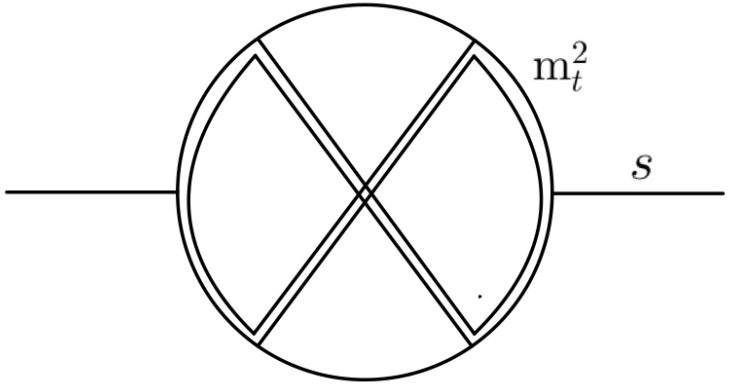
  $$\frac{2.00000000000}{\epsilon^3} + \frac{9.8700393436 + 18.8495559213\ i}{\epsilon^2} - \frac{26.507336797 - 41.196707081\ i}{\epsilon} + (2.29574523 + 201.06880207\ i) + O(\epsilon)$$

- Fully automated following the blueprint Caesar

Few comments about v3t181

- The integral v3t181 has 77 master integrals all in different dimensions, d=4,6,8.
- Automatic resale of master integrals is implemented to meet the requirement that the matrix of system of differential equations is finite in the ϵ series expansion. The largest power is ϵ^{-5}
- The matrix is $\sim$3 MB big before expanding in ϵ

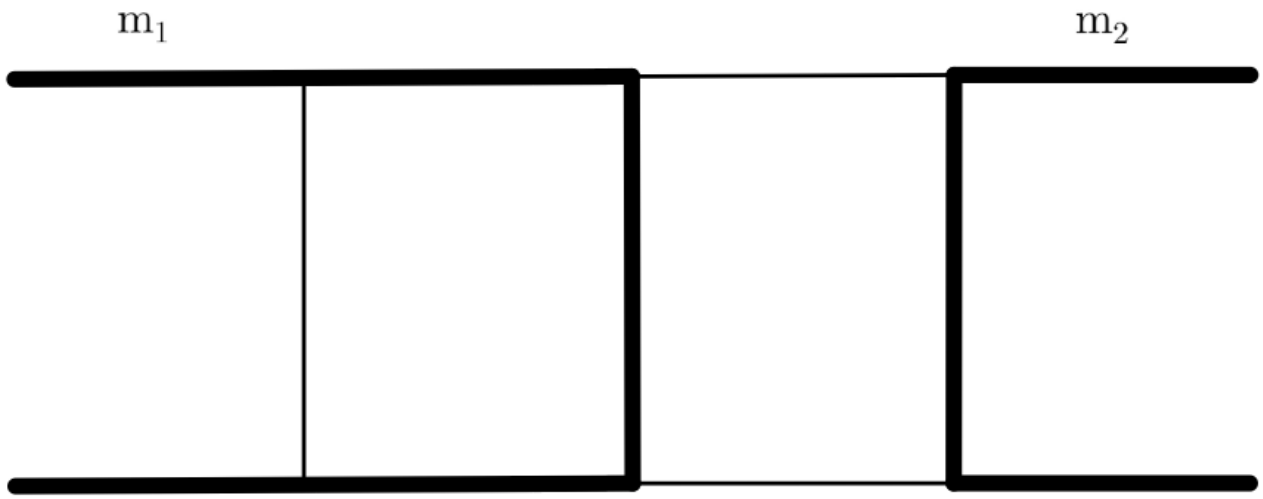
Caesar: Integralfamily taNp1



$$\text{taNp1}^{4-2\epsilon} = \int \frac{[(q_1)^2]^2[(q_2+p_1)^2]}{[(p_1+q_1)^2][(q_1-q_2)^2-m_t^2][(p_1+q_1-q_2)^2-m_t^2]} \frac{\frac{d^Dq_1}{i\pi^{D/2}}\frac{d^Dq_2}{i\pi^{D/2}}\frac{d^Dq_3}{i\pi^{D/2}}}{[(q_2)^2-m_t^2][(q_1-q_3)^2-m_t^2][(q_2-q_3)^2][(q_3)^2-m_t^2][(p_1+q_3)^2-m_t^2]}$$

- In physical regions (s, m_t^2)=(1, $(\frac{433000}{227969})^2$)
  -> $\text{taNp1}^{d=4-2\epsilon} =$
  $8.27490485938[1]/\epsilon^3 - 34.98692810459[0]/\epsilon^2$
  $+102.43077689369[7]/\epsilon - 253.50723525334[2] + O(\epsilon)$
- Fully automated following the blueprint Caesar

Caesar: Integralfamily 2lbox



- In physical regions (s, t, m_1^2, m_2^2)=(2,5,4,16)
  -> 2lbox$^{d=6-2\epsilon}[1,2,1,2,1,1,1,0,0] =$
  $(0.0002973066815 + 0.001542581913\ i)$
  $-(0.002805345908 - 0.003106827180\ i)\ \epsilon + O\left(\epsilon^2\right)$
- Fully automated following the blueprint Caesar

Caesar: Iterative Approach

- Generate a system of differential equations for just one variable s_1 and set all other kinematic variables to numeric values $s_i = n_i$(rational number), $i \neq 1$
- The integration-by-parts reductions depend only on s_1 and d
- Evaluate all master integrals in a comfortable point numerically with pySecDec or AMFlow (we call pySecDec or AMFlow only once)
- Transport the boundary terms with DiffExp to some useful value u_1 in the physical regions for s_1
- Generate a system of differential equations for the next variable s_2 and set all other kinematic variables to numeric values $s_i = n_i$(rational number), $i \neq 1, 2$ and $s_1 = u_1$(physical region)
- We skip the evaluation with pySecDec, since we know the new boundary terms from the last DiffExp call
- Transport the boundary terms with DiffExp to the next useful physical value u_2 for s_2
- Continue with this pattern

Applications of the Iterative Approach

- Build a grid out of lines (generated with Caesar) for numerical integration algorithms
- IBP reductions are cheap, but for each new point outside the line new reduction is needed $\rightarrow$ smart presampling is needed
- Improve Kira, we need the block triangular form implementation in cpp.

Outlook

- Get a basis where the matrix of the system of differential equations is linear in ϵ
  - -> DiffExp does order of magnitudes faster transport of the boundary terms
- Iterative application of the blueprint Caesar
- Generate grids based on lines produces with Caesar
- Kira supports deformed propagators
- Implement automated search for Euclidean regions

Conclusions

- The first physics goals are already in next month reach
- Important is the knowledge transfer and to get people motivated to engineer other methods for practical applications
- Strong computing resources are needed not only for the final product but also for the development of the tools.
- Without spending significant effort on simplification of the basis, we can numerically solve the differential equations of non-trivial 3-loop Feynman integrals.
- By choosing the basis representatives to be finite integrals, we can obtain precise numerical boundary conditions in the Euclidean region using pySecDec or AMFlow.
- We find that the precision of the boundary conditions in the Euclidean region carries over to the physical region.
- The process is fully automated.
- The list of applications possible in physics with Caesar is growing.