Status of the Gepard code

Krešimir Kumerički

University of Zagreb

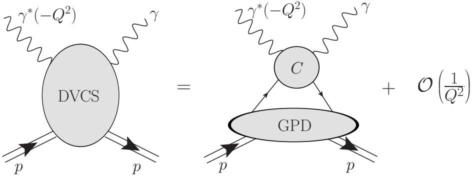
Joint GDR-QCD/QCD@short distances and STRONG2020/PARTONS/FTE@LHC/NLOAccess workshop
31 May to 4 June 10, 2021

Hrzz Croatian Science Foundation
EUROPSKI STRUKTURNI I INVESTICIJSKI FONDOVI
STRONG-2020
HRT

Outline

Factorization of DVCS $\longrightarrow$ GPDs

- [Collins et al. '98]



- CFFs are convolution:

$$ {}^{a}\mathcal{H}(\xi, t, Q^2) = \int \mathrm{d}x \; T^a\Big(x, \xi, \frac{Q^2}{Q_0^2}\Big) \, H^a(x, \xi, t, Q_0^2) $$

$$ a = q, G $$

- $H^a(x, \eta, t, Q_0^2)$ — Generalized parton distribution (GPD)

Modelling GPDs in conformal moment space

- Instead of considering momentum fraction dependence $H(x, \dots)$
- …it is convenient to make a transform into complementary space of conformal moments j:

$$H^q_j(\eta, \dots) \equiv \frac{\Gamma(3/2)\Gamma(j+1)}{2^{j+1}\Gamma(j+3/2)} \int_{-1}^{1} \mathrm{d}x \, \eta^j \, C_j^{3/2}(x/\eta) \, H^q(x, \eta, \dots)$$

- They are analogous to Mellin moments in DIS: $x^j \to C_j^{3/2}(x)$
- $C_j^{3/2}(x)$ — Gegenbauer polynomials

CFFs as Mellin-Barnes integral

$$\mathcal{H}(\xi, t, Q^2) = \frac{1}{2i} \int_{c-i\infty}^{c+i\infty} dj \, \xi^{-j-1} \left[i + \tan\left(\frac{\pi j}{2}\right) \right]$$

$$\times \; T_j(Q^2/\mu^2, \alpha_s(\mu)) \; H_j(\xi, t, \mu^2) \, .$$

- Evolution of GPDs:

$$H_j(\eta, t, \mu) = \sum_k E_{jk}(\mu, \mu_0; \eta) \; H_k(\eta, t, \mu_0) \; ,$$

- T_j and E_{jk} known to (N)NLO
- $\sum_k$ is infinite $\rightarrow$ resummation leads to second Mellin-Barnes integral

Advantages of conformal space

1. The evolution equations are most simple: There is **no mixing** among moments at LO, and in special ($\overline{CS}$) scheme not even at NLO
2. Powerful analytic methods of **complex *j*** plane are available (similar to complex angular momentum of Regge theory)
3. Stable and fast **computer code** for evolution and fitting
4. Moments are equal to matrix elements of **local** operators and are thus directly accessible on the **lattice**

Disadvantages of conformal space

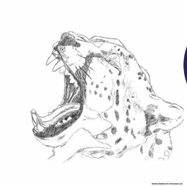
1. Difficult conversion between x-space and j-space. E.g. [Müller and Schäfer '05] toy model:

$$H^{\text{toy}}(x, \eta) = \frac{1}{1-\alpha}\theta(-\eta \leq x)\,\frac{1}{\eta}\left(\frac{x+\eta}{1+\eta}\right)^{1-\alpha}$$

$$H_j^{\text{toy}}(\eta) \quad \propto \quad \frac{(1+j-\alpha)(2+j-\alpha)}{2(1-\alpha)(2-\alpha)}\,\frac{\Gamma(3/2)\Gamma(3+j)}{\Gamma(3/2+j)} \times \frac{\eta-1}{2\eta}\left(\frac{\eta}{2}\right)^j {}_3F_2\left(\left.\begin{matrix}-j, 3+j, 2-\alpha\\ 2, 3-\alpha\end{matrix}\,\right|\,\frac{\eta-1}{2\eta}\right)$$

2. Need complex-analytic continuation of functions used for modelling. How to work with numerical models, neural nets?

Gepard* software



Gepard

*Not to be confused with biology software (`GEnome PAir Rapid Dotter`)

Gepard — features

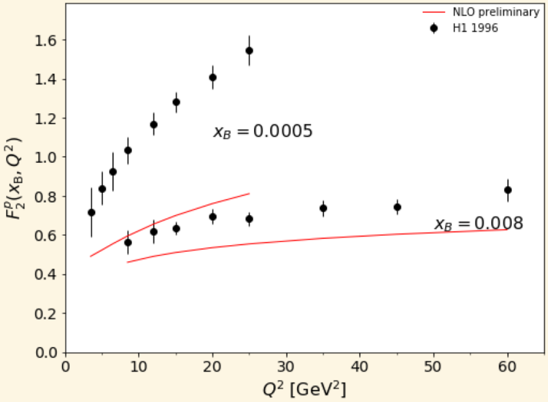
- Gepard implements [Belitsky, Müller et al.] DVCS formulas for all measured DVCS observables
- Implemented GPD/CFF models
  - Goloskokov-Kroll (GK)
  - K.K. and D. Müller (KM)
  - Neural networks
- Fitting procedures
  - MINUIT least-squares fitting
  - Neural networks (PyBrain and TensorFlow (experimental))
- interactive work possible (jupyter notebooks)

GeParD — interactive example 1/7

```
In [12]: m = Model.ComptonGepard(p=1, q02=2.)      # initialize NLO KM model for CFFs
         th = Approach.BMK(m)                       # use BMK formulas for DVCS
         th.name = 'NLO preliminary'                # name to be used on plots

In [13]: th.m.parameters['AL0S'] = 1.2              # change some model parameter

In [14]: utils.describe_data(DISpoints)             # DISpoints = data set

         npt x obs    collab  FTn    id  ref.
         ----------------------------------------------
          8 x F2      H1      N/A    201 Nucl.Phys.B470(96)3
          8 x F2      H1      N/A    202 Nucl.Phys.B470(96)3
          9 x F2      H1      N/A    203 Nucl.Phys.B470(96)3
          9 x F2      H1      N/A    204 Nucl.Phys.B470(96)3
```

GeParD — interactive example 2/7

```
In [41]: fig = plots.HERAF2Q2(lines=[th])   # specificaly designed plots
```




```
In [18]: th.print_chisq(DISpoints)

P(chi-square, d.o.f) = P(3787.21, 85) = 0.0000
```

GeParD — interactive example 3/7

In [19]:
```
%%time
th.model.release_parameters('NS', 'AL0S', 'AL0G')
f = Fitter.FitterMinuit(DISpoints, th)
f.fit()
```

FCN = 72.85271021975446	TOTAL NCALL = 115	NCALLS = 115
EDM = 1.082463486221191e-06	GOAL EDM = 1e-05	UP = 1.0

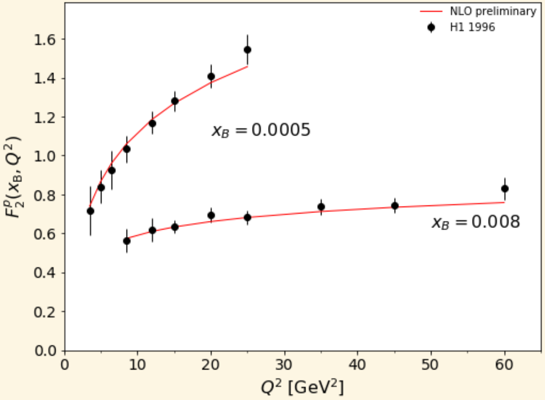
Valid	Valid Param	Accurate Covar	PosDef	Made PosDef
True	True	True	True	False
Hesse Fail	HasCov	Above EDM		Reach calllim
False	True	False		False

±	Name	Value	Hesse Error	Minos Error-	Minos Error+	Limit-	Limit+	Fixed?
0	NS	0.149507	0.00706756					No
1	AL0S	1.06274	0.0197111					No
2	ALPS	0.15	1					Yes

- Two “fitters” are implemented: Minuit and Neural network

GeParD — interactive example 4/7

```
In [20]: th.print_chisq(DISpoints)

P(chi-square, d.o.f) = P(72.85, 82) = 0.7549

In [21]: fig = plots.HERAF2Q2(lines=[th])
```

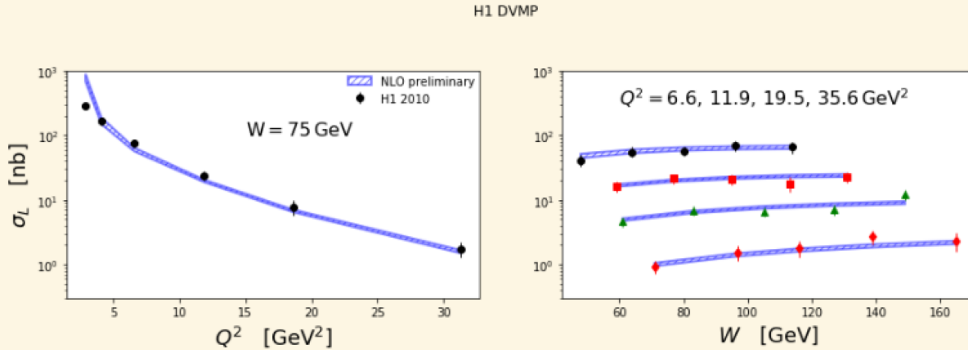


GeParD — interactive example 5/7

```
In [22]: utils.describe_data(DVMPpoints)

         npt x obs    collab  FTn    id  ref.
         ---------------------------------------------
          5 x X      H1      N/A    76  arXiv:0910.5831
         20 x X      H1      N/A    79  arXiv:0910.5831
         ---------------------------------------------
         TOTAL = 25

Out[22]: 25

In [23]: th.print_chisq(DVMPpoints)

         P(chi-square, d.o.f) = P(1752.85, 22) = 0.0000

In [24]: th.model.fix_parameters('NS', 'AL0S', 'AL0G')
         th.model.release_parameters('M02S', 'M02G', 'SECS', 'SECG', 'THIS', 'THIG')

In [25]: %%time
         f = Fitter.FitterMinuit(DVMPpoints, th)
         f.fit()

         CPU times: user 9min 55s, sys: 582 ms, total: 9min 55s
         Wall time: 25.7 s

In [30]: th.print_chisq(DISpoints+DVMPpoints)

         P(chi-square, d.o.f) = P(85.51, 104) = 0.9066
```

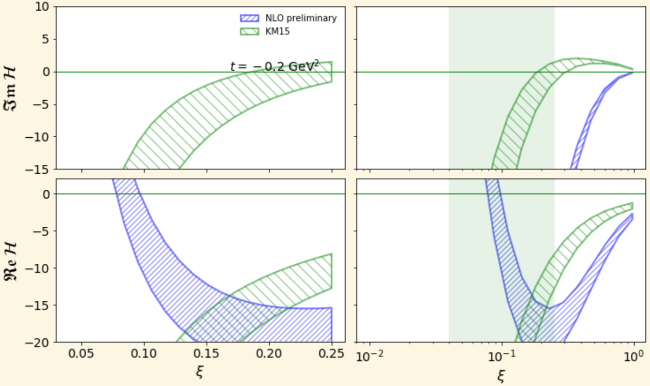
GeParD — interactive example 6/7

```
fig = plots.DVMP(wdep, bands=[th])
```



GeParD — interactive example 7/7

```
In [33]: fig = plots.CFF(bands=[th, KM15])
```



Code rewrite

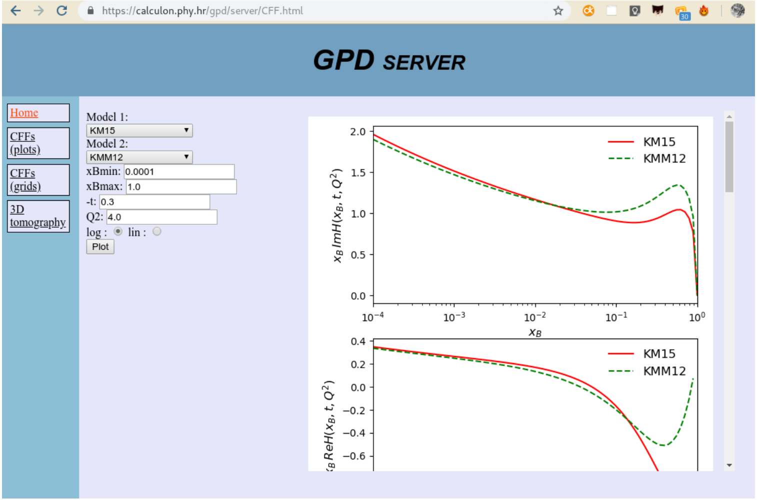
Code is presently in the transition from this:

Language	files	blank	comment	code
Python	46	2289	2865	13061
Fortran 77	120	7822	30240	12190
IPython Notebook	2	0	0	1956
C	5	161	263	454
Mathematica	1	124	15	415
make	4	101	133	296
C/C++ Header	2	40	74	91
DOS Batch	1	0	0	44
Markdown	1	3	0	5
SUM:	182	10540	33590	28512

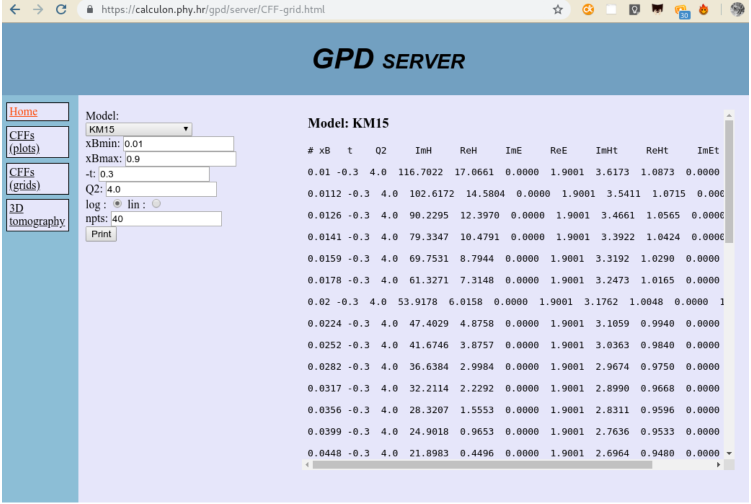
to this:

Language	files	blank	comment	code
Python	17	1426	1581	5392
SUM:	17	1426	1581	5392

GPD/CFF server



GPD/CFF server

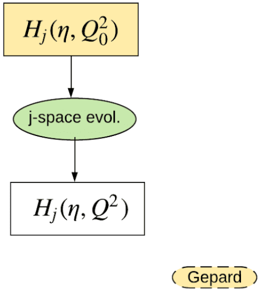


# xB	t	Q2	ImH	ReH	ImE	ReE	ImHt	ReHt	ImEt	
0.01	-0.3	4.0	116.7022	17.0661	0.0000	1.9001	3.6173	1.0873	0.0000	
0.0112	-0.3	4.0	102.6172	14.5804	0.0000	1.9001	3.5411	1.0715	0.000	
0.0126	-0.3	4.0	90.2295	12.3970	0.0000	1.9001	3.4661	1.0565	0.0000	
0.0141	-0.3	4.0	79.3347	10.4791	0.0000	1.9001	3.3922	1.0424	0.0000	
0.0159	-0.3	4.0	69.7531	8.7944	0.0000	1.9001	3.3192	1.0290	0.0000	
0.0178	-0.3	4.0	61.3271	7.3148	0.0000	1.9001	3.2473	1.0165	0.0000	
0.02	-0.3	4.0	53.9178	6.0158	0.0000	1.9001	3.1762	1.0048	0.0000	1
0.0224	-0.3	4.0	47.4029	4.8758	0.0000	1.9001	3.1059	0.9940	0.0000	
0.0252	-0.3	4.0	41.6746	3.8757	0.0000	1.9001	3.0363	0.9840	0.0000	
0.0282	-0.3	4.0	36.6384	2.9984	0.0000	1.9001	2.9674	0.9750	0.0000	
0.0317	-0.3	4.0	32.2114	2.2292	0.0000	1.9001	2.8990	0.9668	0.0000	
0.0356	-0.3	4.0	28.3207	1.5553	0.0000	1.9001	2.8311	0.9596	0.0000	
0.0399	-0.3	4.0	24.9018	0.9653	0.0000	1.9001	2.7636	0.9533	0.0000	
0.0448	-0.3	4.0	21.8983	0.4496	0.0000	1.9001	2.6964	0.9480	0.0000	

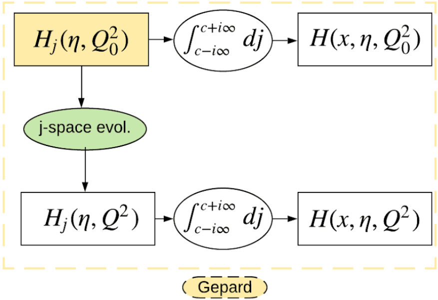
- Gepard code itself is on `github.com`, but not public yet

Checks and benchmarks

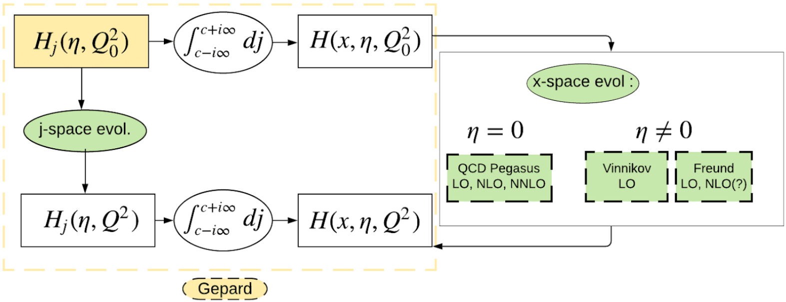
Testing evolution



Testing evolution

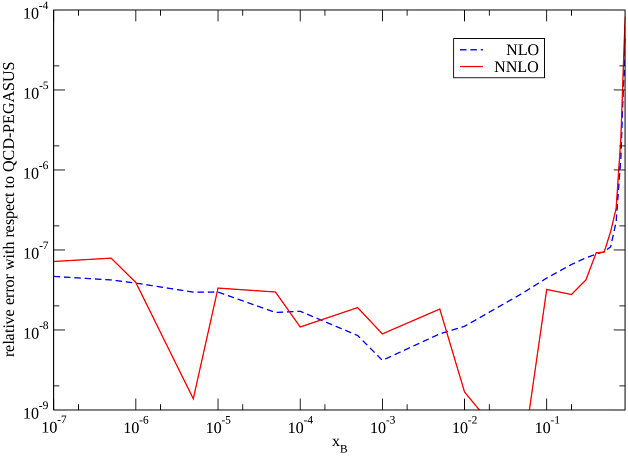


Testing evolution



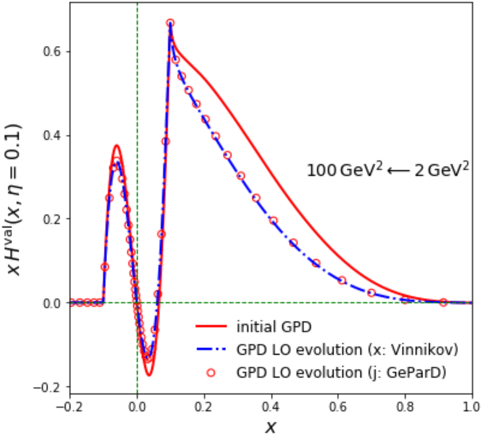
Checking evolution in forward limit

- Comparison to QCD-Pegasus PDF evolution software [A. Vogt '04]



Checking LO evolution

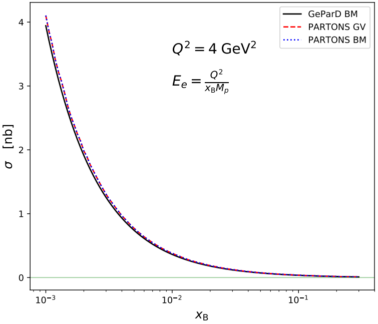
- Comparison to GPD evolution software [Vinnikov '06]



- Maximal relative discrepancy: 2 %.

Gepard vs PARTONS, cross-section

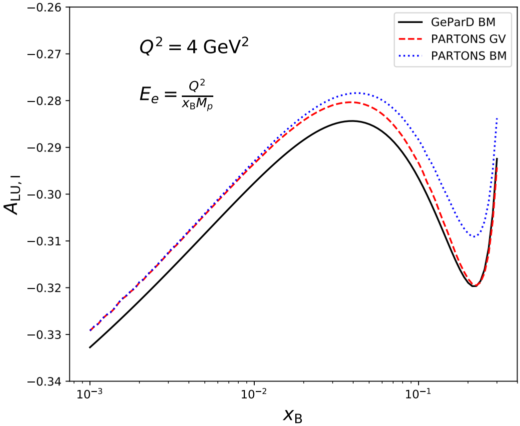
- BM = [Belitsky & Müller], GV = [Guichon & Vanderhaeghen]



- 100 kinematical points:
  - Gepard (python): 24 seconds
  - PARTONS (C++, XML driven): 9 seconds

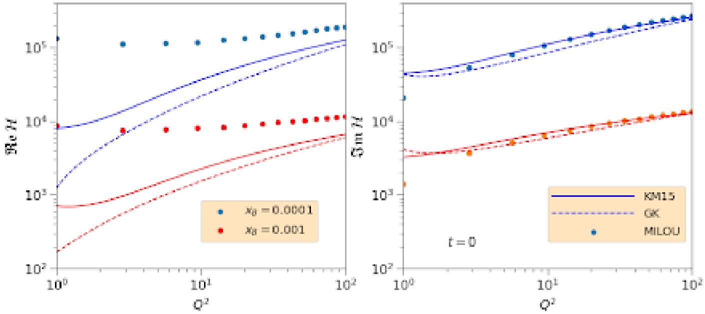
Gepard vs PARTONS, beam spin asymmetry

- BM = [Belitsky & Müller], GV = [Guichon & Vanderhaeghen]



Comparison of popular models

- GK = [Goloskokov & Kroll]
- KM = [K.K. & Müller]
- MILOU = DVCS MC generator, model by [Freund & McDermott]



Outlook

- Implementation of DVMP — work in progress
- Going from hybrid to full conformal-space GPD model — work in progress
- All the components for the NLO analysis are available — work in progress

The End