Minimum Error Training of Log-Linear Translation Models

Mauro Cettolo & Marcello Federico

ITC-irst - Centro per la Ricerca Scientifica e Tecnologica
I-38050 Povo (Trento), Italy

{cettolo,federico}@itc.it

Overview

- Log-Linear Models for MT
- Minimum Error Training
- Simplex Algorithm
- Experimental Results
- Conclusions

Log-linear Models in ASR and SMT

Log-linear models were introduced in ASR by Philips labs in the late '90s:

- log-linear interpolation of language models [Klakow, 1998]
- scaling factor estimation to minimize recognition errors [Beyerlein, 1997]

More recently, log-linear models have been introduced in SMT:

- maximum entropy models and discriminative training for SMT [Och & Ney, 2002]
- minimum error rate training in SMT [Och, 2003].

Our work is related to [Och, 2003], but investigates a different training technique.

Maximum Entropy Framework for SMT

Maximum Entropy framework for word-alignment MT approach:

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} \sum_{\mathbf{a}} \Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) \approx \arg\max_{\mathbf{e}} \max_{\mathbf{a}} \Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) \quad (1)$$

$\Pr(\mathbf{e}, \mathbf{a} \mid \mathbf{f})$ **is determined through real valued feature functions** $h_i(\mathbf{e}, \mathbf{f}, \mathbf{a}), i = 1 \ldots M$**, and takes the parametric form:**

$$p_\lambda(\mathbf{e}, \mathbf{a} \mid \mathbf{f}) = \frac{\exp\{\sum_i \lambda_i h_i(\mathbf{e}, \mathbf{f}, \mathbf{a})\}}{\sum_{\mathbf{e},a} \exp\{\sum_i \lambda_i h_i(\mathbf{e}, \mathbf{f}, \mathbf{a})\}} \quad (2)$$

Example: feature functions of IBM Model 4:

$$\begin{array}{lcll} h_1(\mathbf{e}, \mathbf{f}, \mathbf{a}) & = & \log \Pr(\mathbf{e}) & \text{(target language model)} \\ h_2(\mathbf{e}, \mathbf{f}, \mathbf{a}) & = & \log \Pr(\phi \mid \mathbf{e}) & \text{(fertility model)} \\ h_3(\mathbf{e}, \mathbf{f}, \mathbf{a}) & = & \log \Pr(\tau \mid \mathbf{e}, \phi) & \text{(lexicon model)} \\ h_4(\mathbf{e}, \mathbf{f}, \mathbf{a}) & = & \log \Pr(\pi \mid \mathbf{e}, \phi, \tau) & \text{(distortion model)} \end{array}$$

Search Criterion and Properties

The search criterion of MT can be rewritten as:

$$\mathbf{e}^* = \arg\max_{\mathbf{e}} \max_{\mathbf{a}} \sum_i \lambda_i h_i(\mathbf{e}, \mathbf{f}, \mathbf{a})\} \quad \textbf{(3)}$$

The ME framework gives the following advantages:

- **directly models the posterior probability (discriminative model)**
- **does not rely on probability factorizations with independence assumptions**
- **its mathematically sound framework permits to add any kind of feature**
- **includes any IBM-model as special case, e.g. see previous slide with λ set to 1**
- **ML or minimum error training can be applied to estimate free parameters (λ)**

Training of Log-Linear Models

Instead of applying MLE, training can directly address performance optimization:

$$\lambda_* = \arg\min_{\lambda} E_D(\lambda) \quad (4)$$

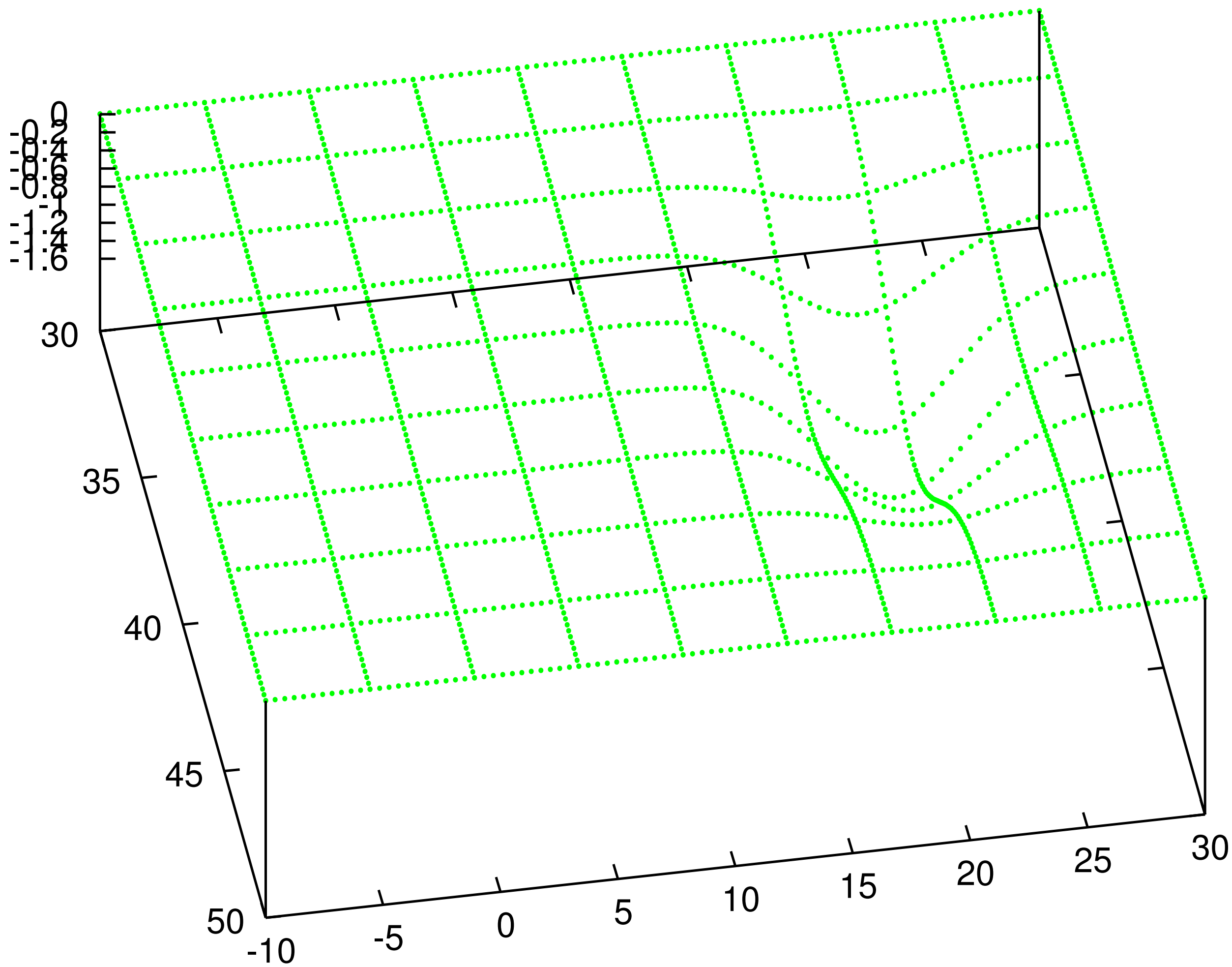
where $E_D(\lambda)$:

- measures translation errors over a development set D, e.g. Bleu, Nist, WER, PER
- can be very irregular, i.e. has many local minima

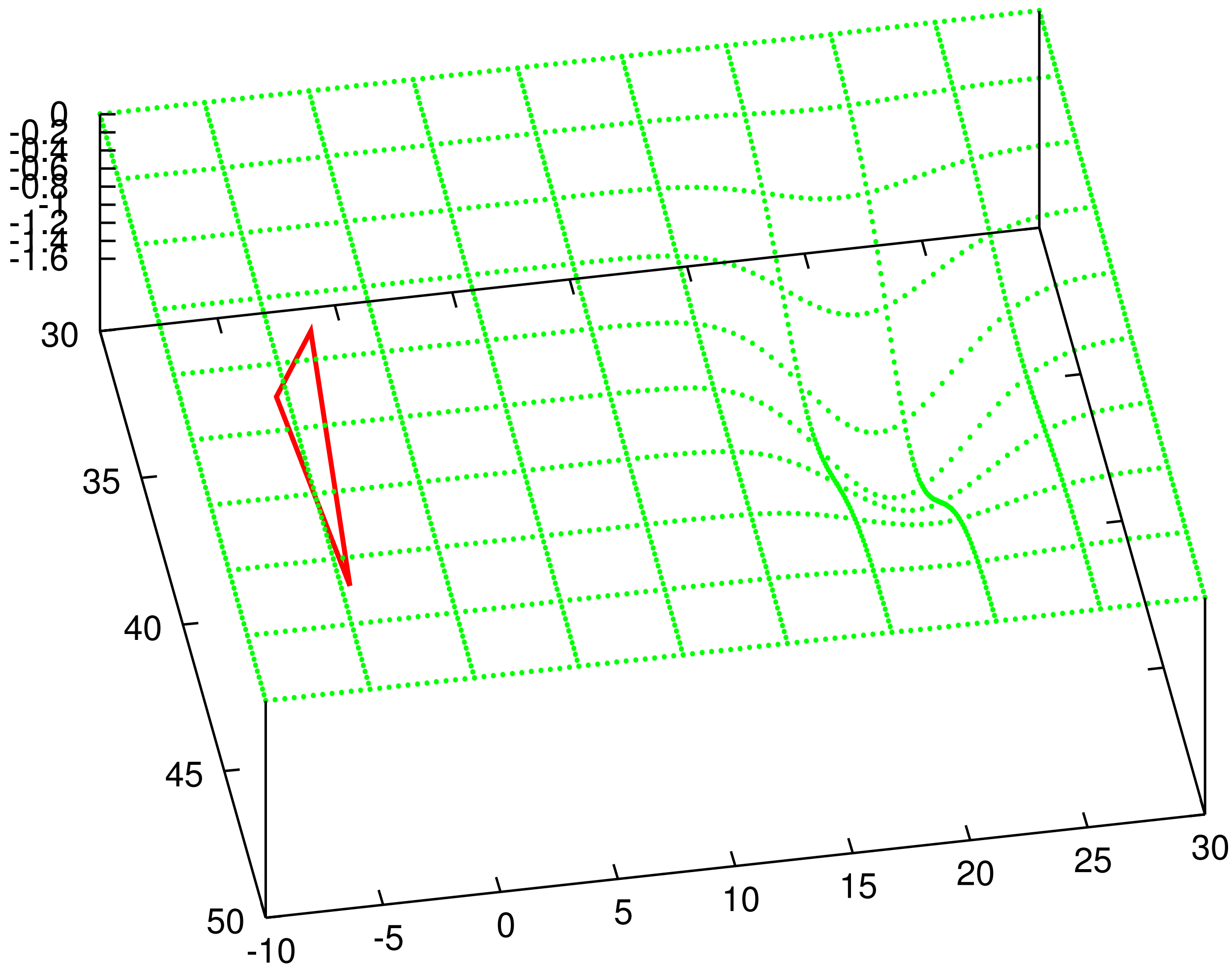
We apply a multi-variate minimization algorithm, called simplex, which:

- empirically evaluates $E_D(\lambda)$ several times until convergence
- requires running the SMT search algorithm for each evaluation

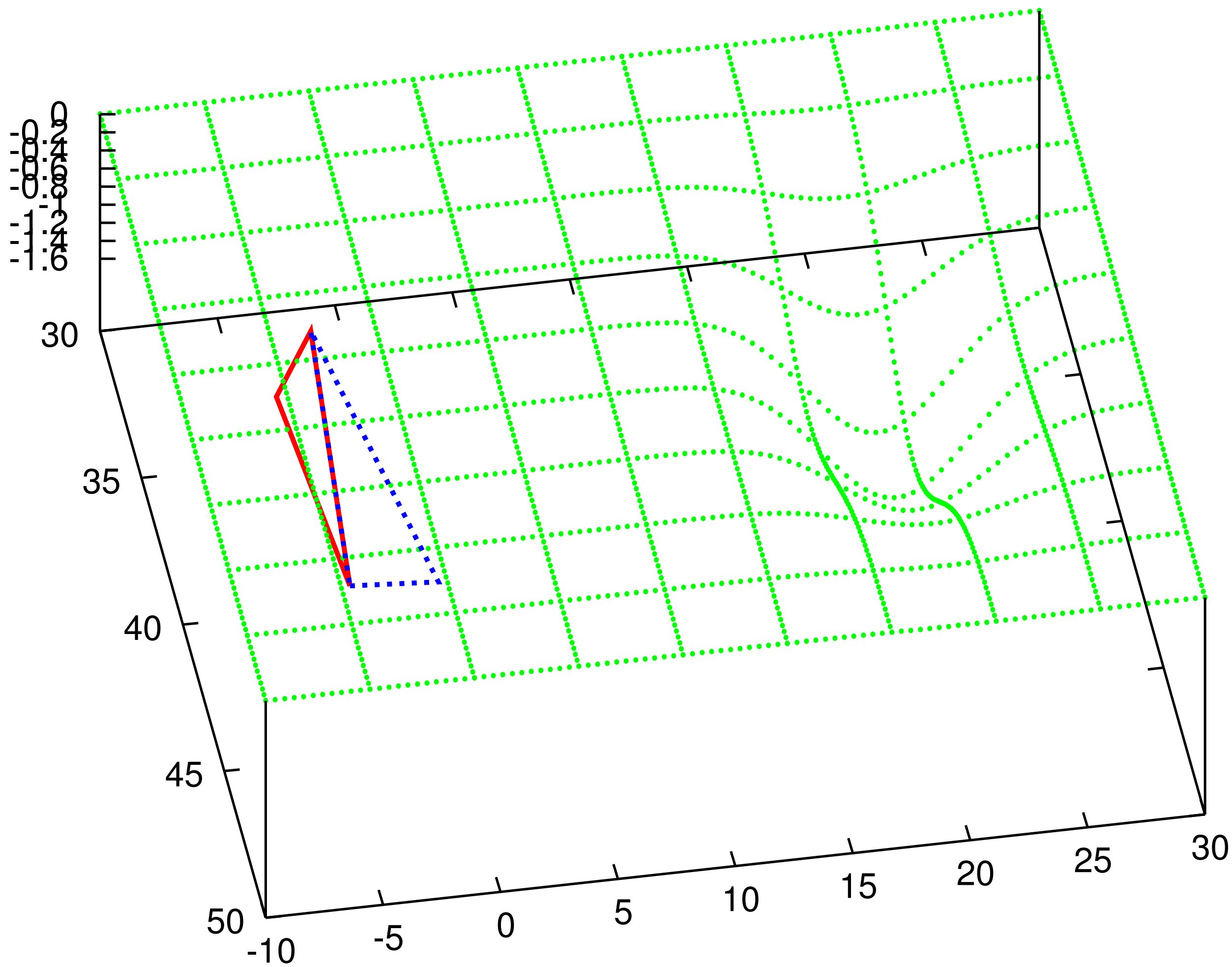
The same approach was independently applied by [Zens & Ney, 2004]



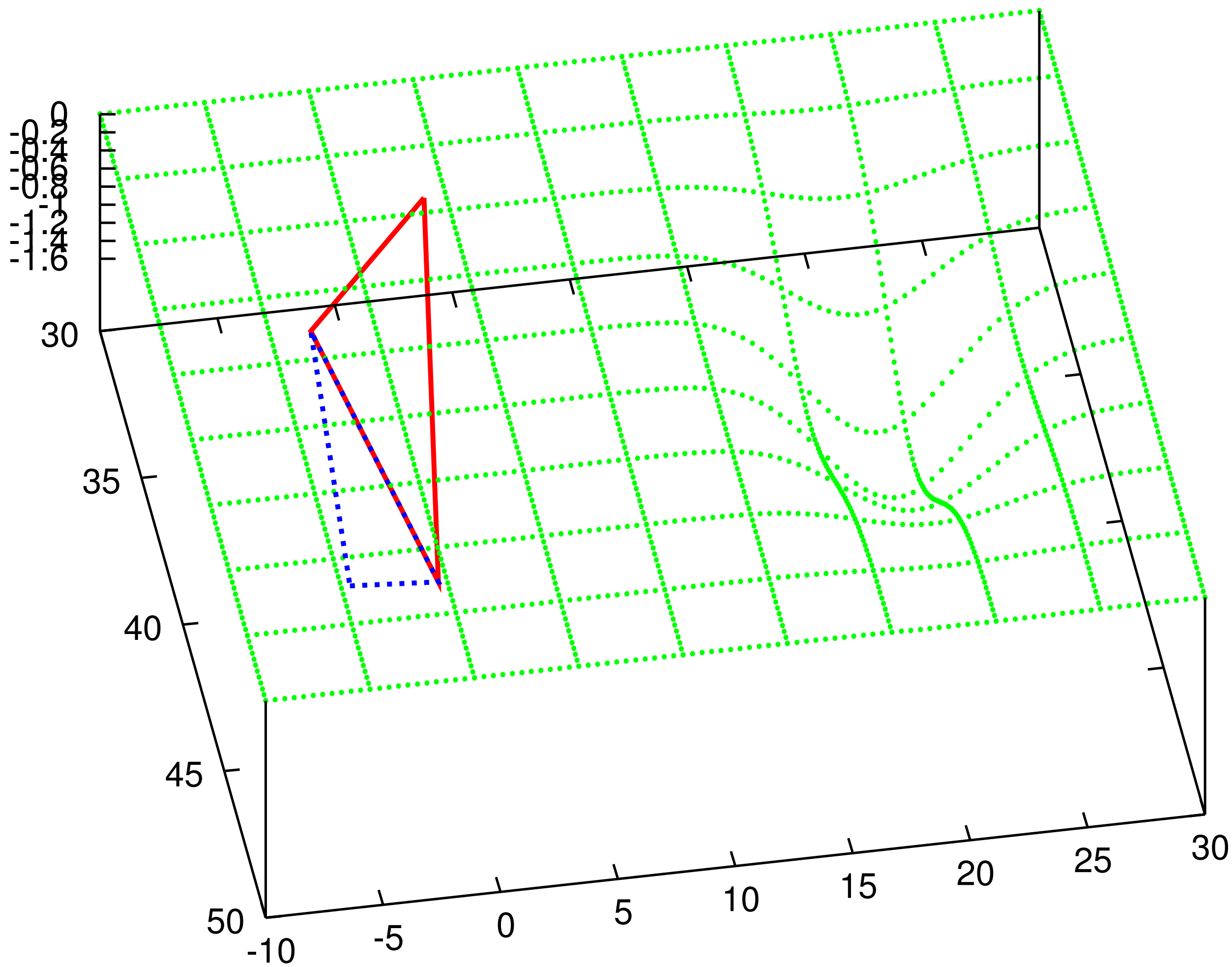
0
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30



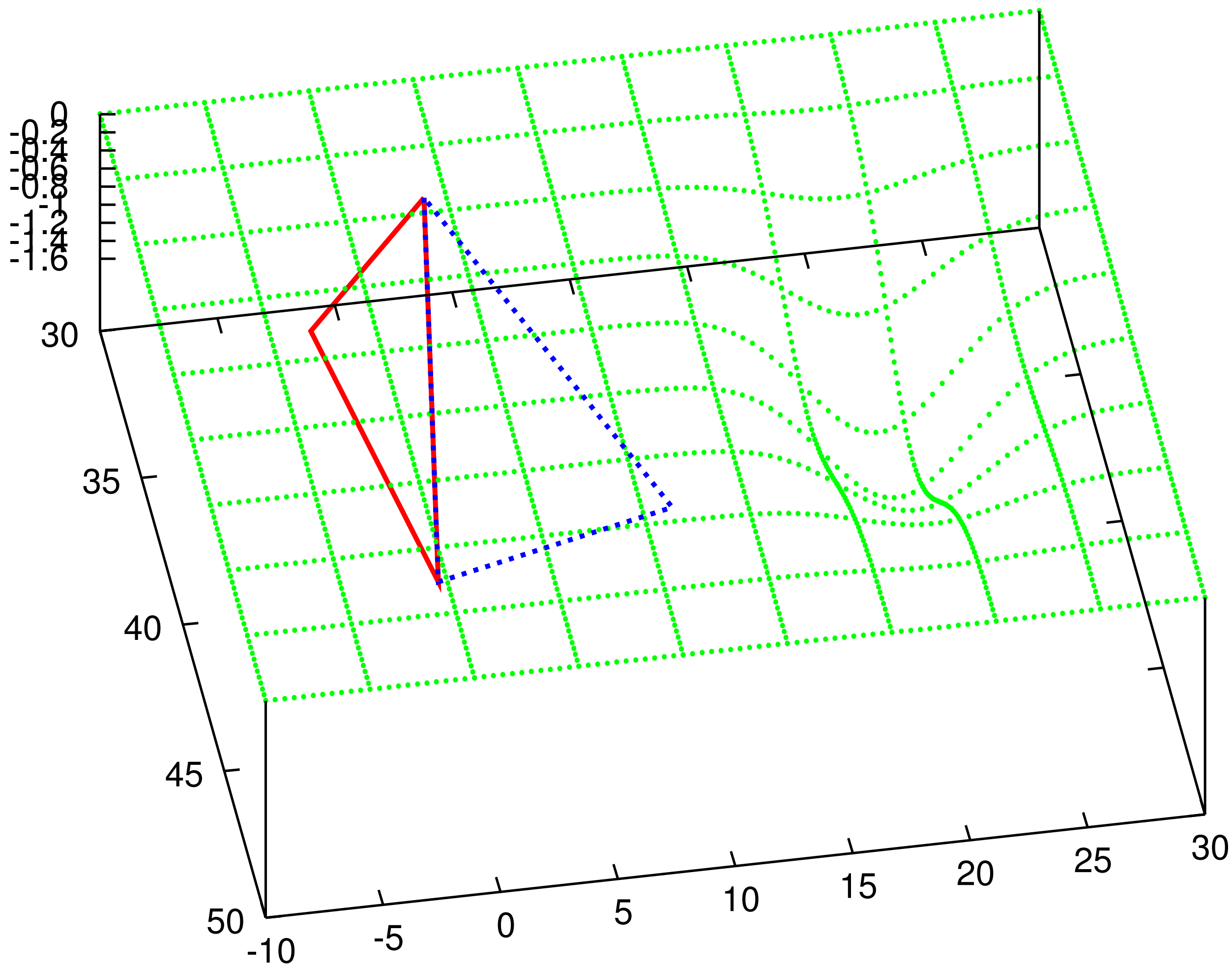
0
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30



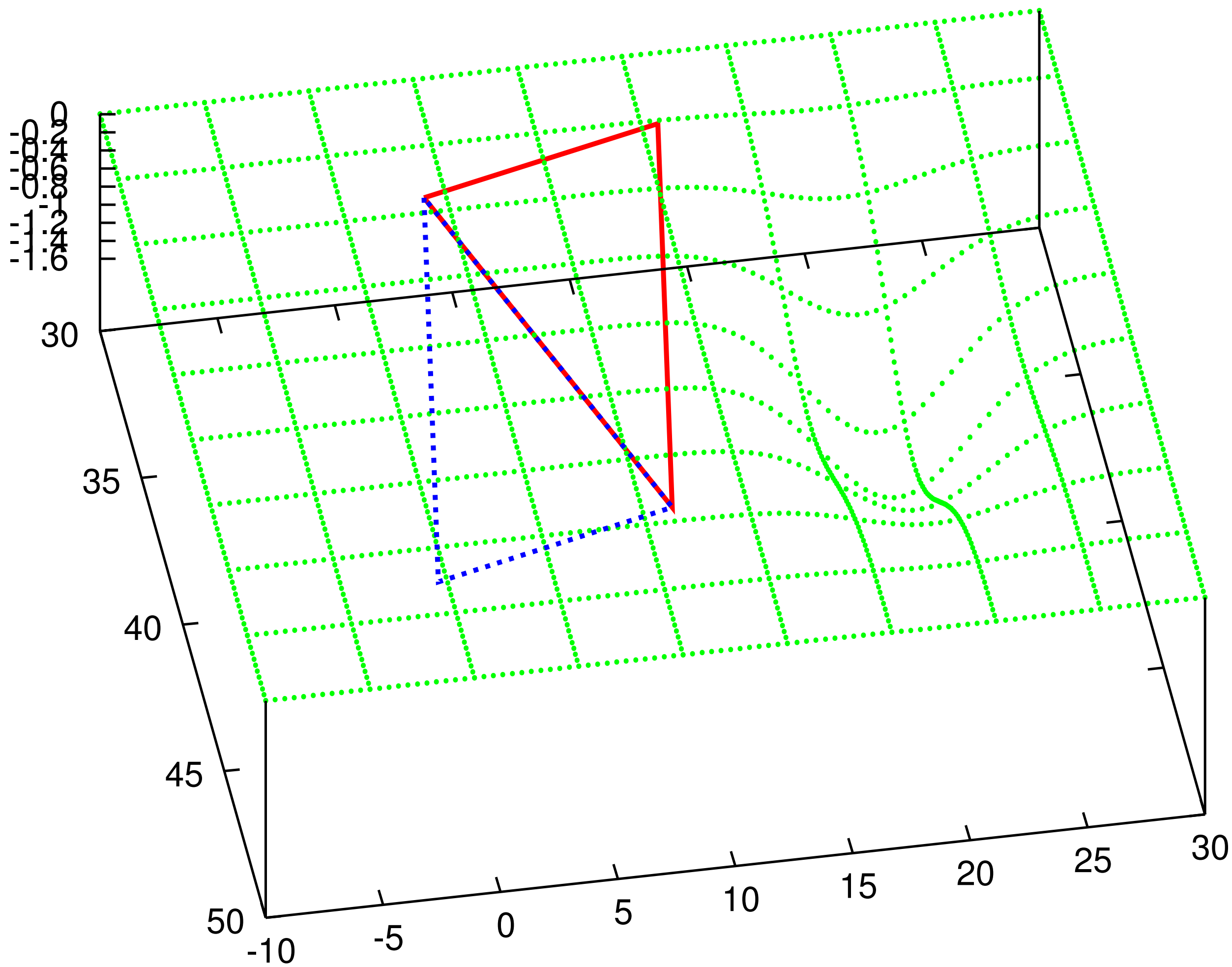
0
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

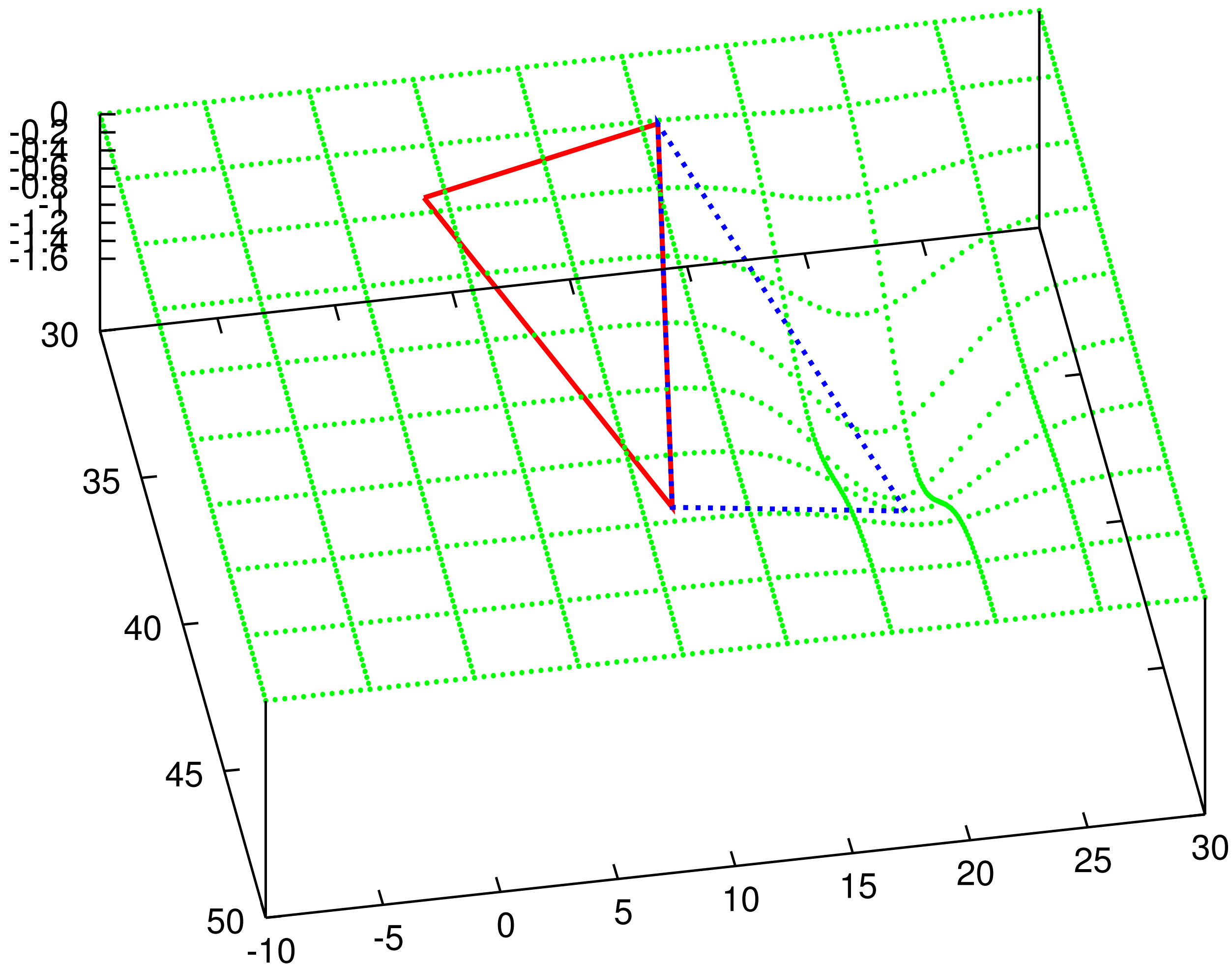


0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

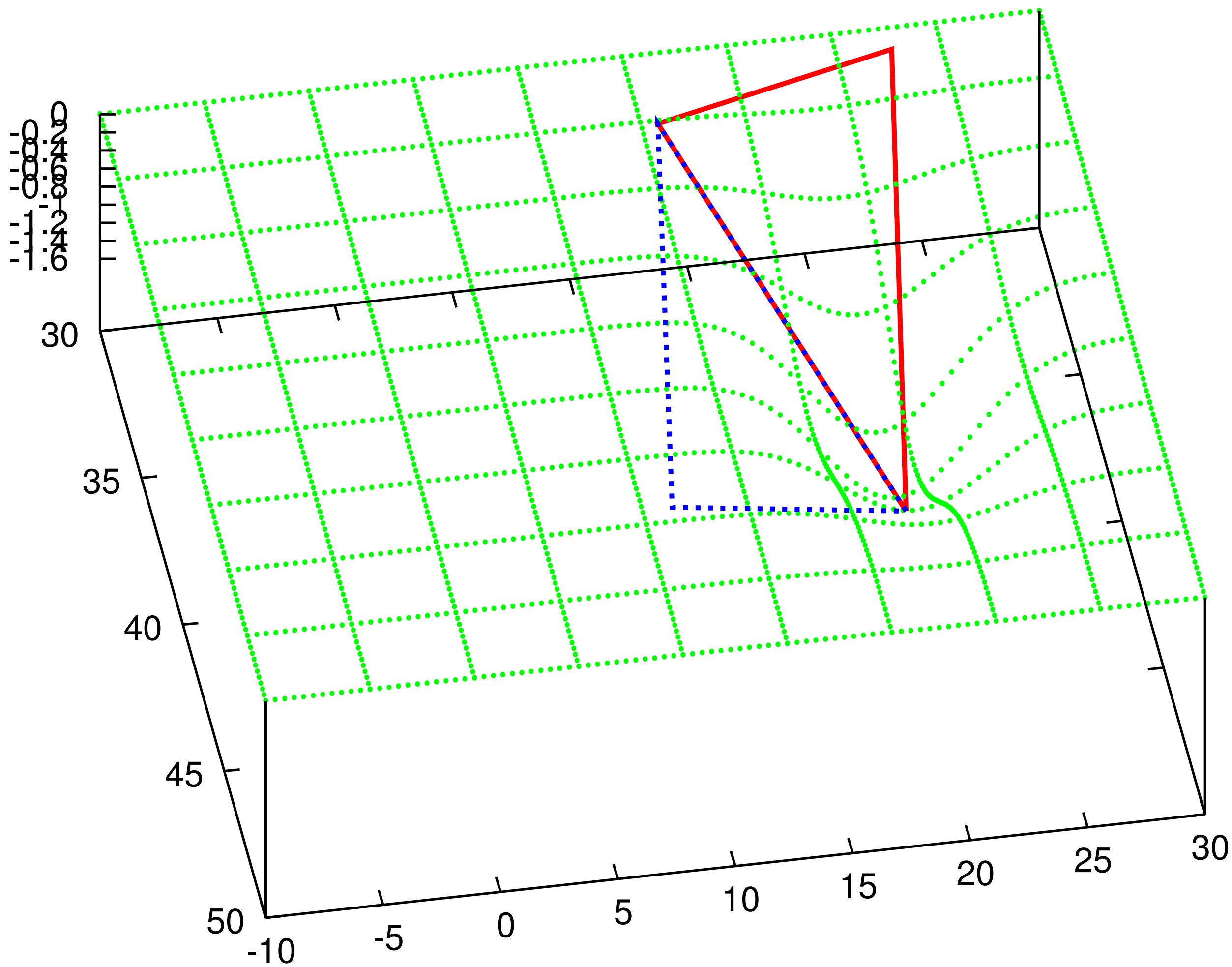


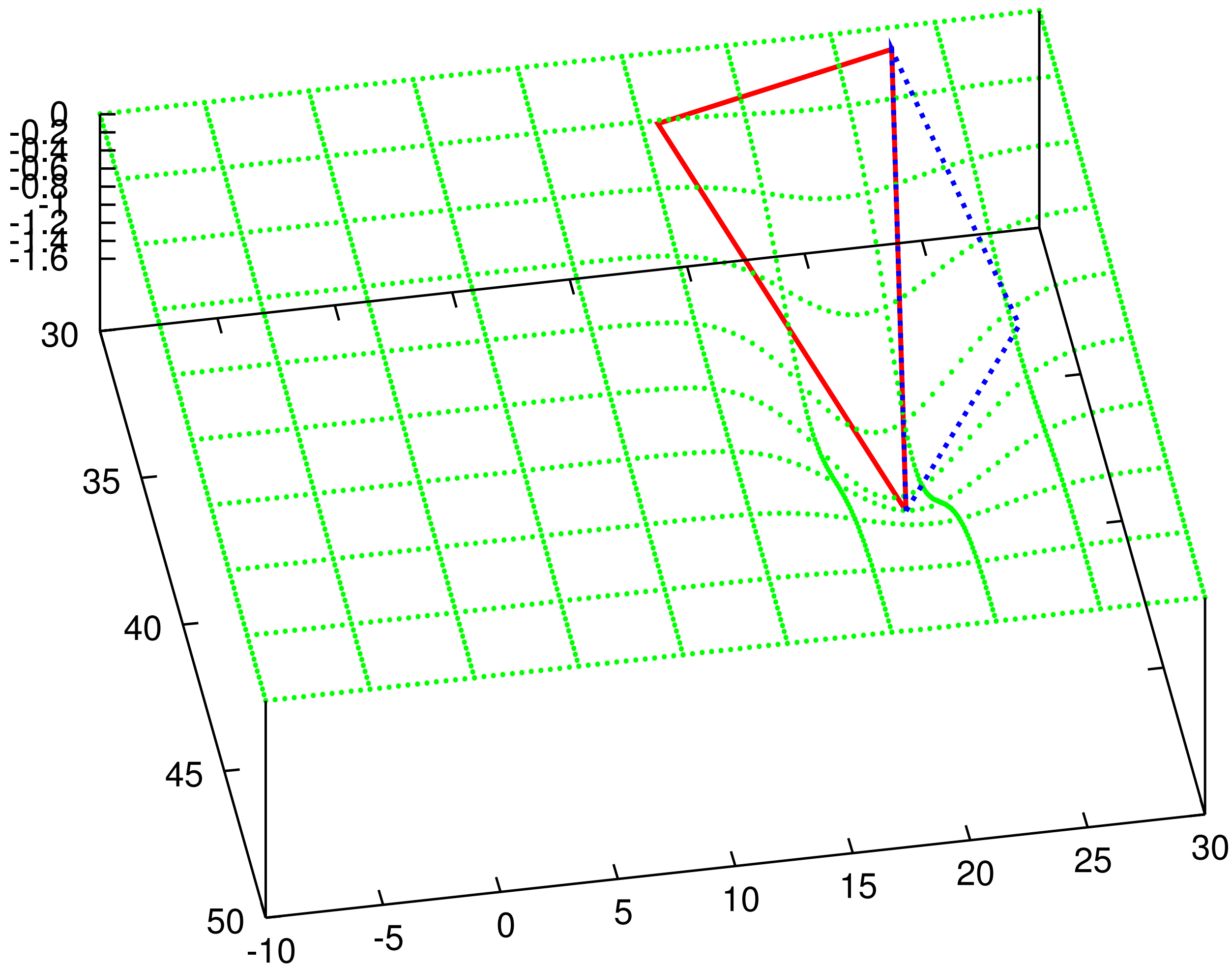
0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

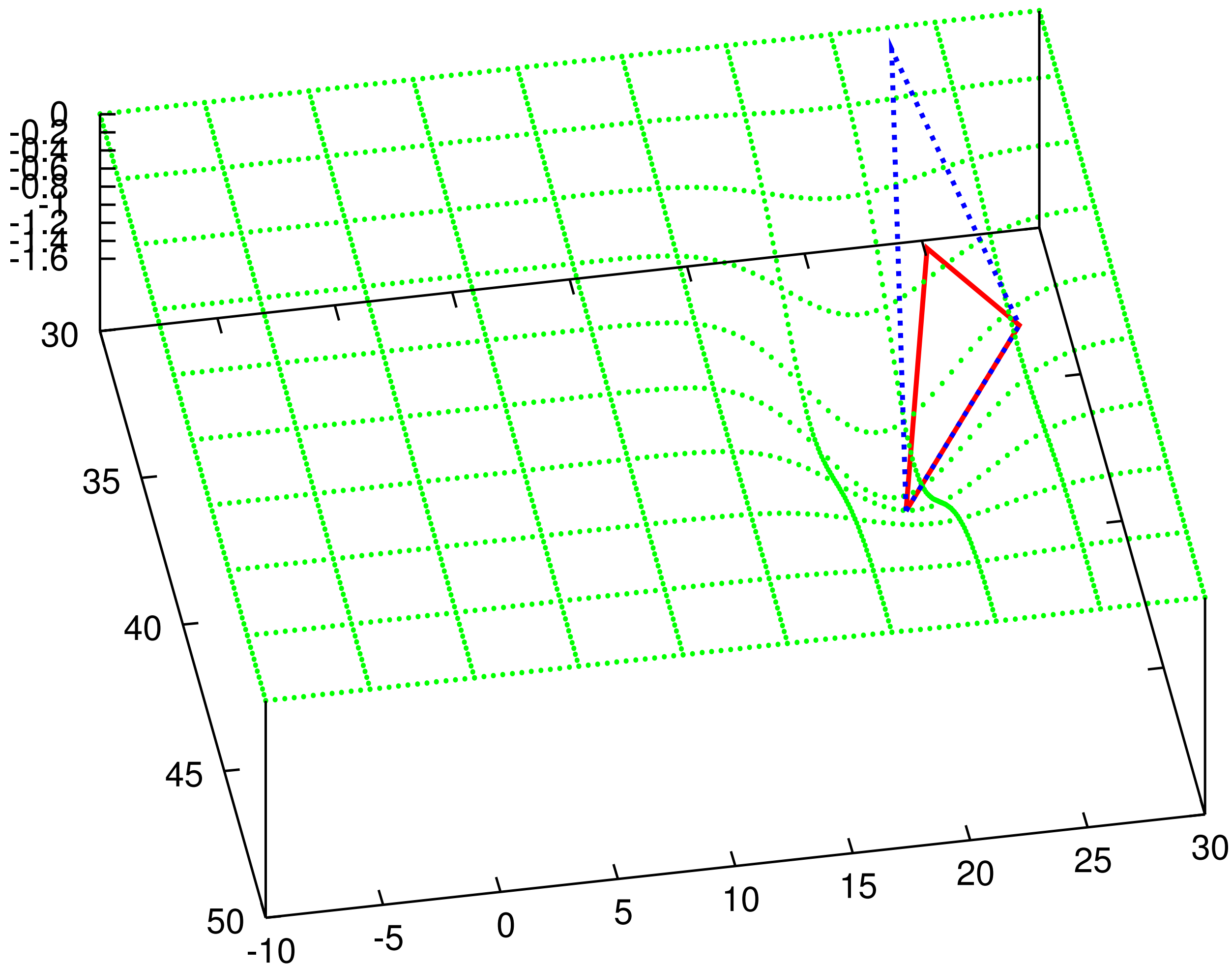

0
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

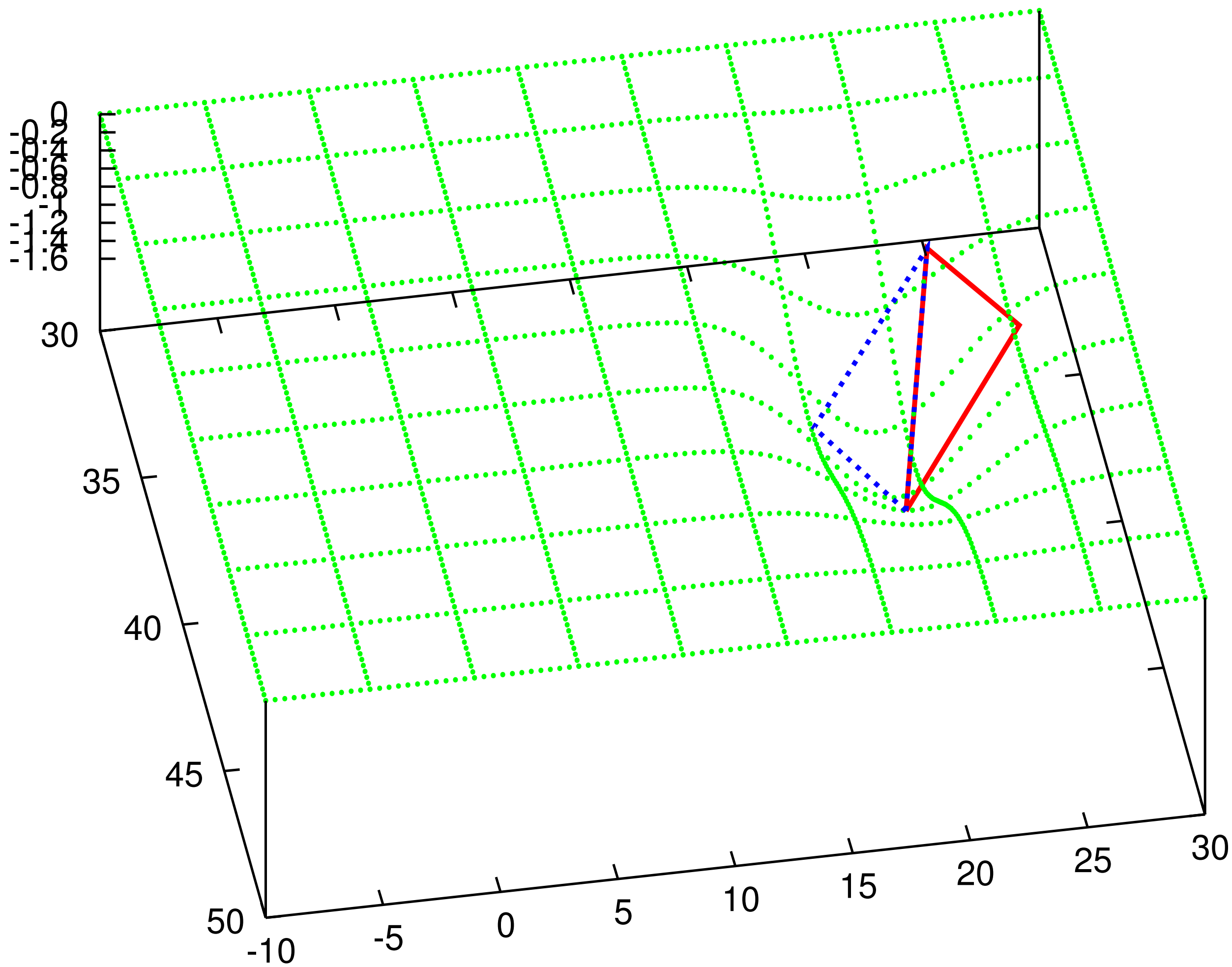


0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

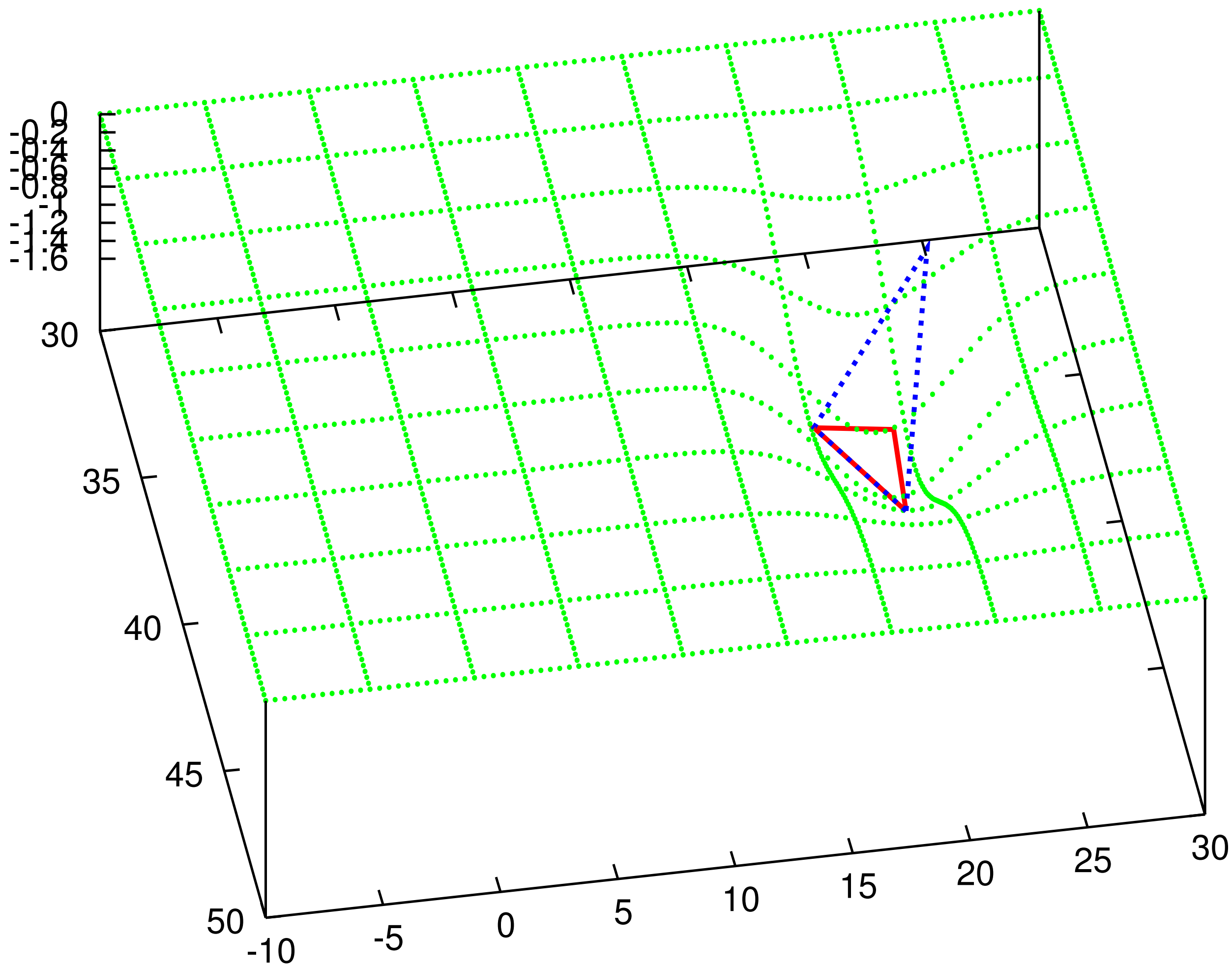

0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30


0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30


0
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30



0
-0.2
-0.4
-0.6
-0.8
-1
-1.2
-1.4
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30


0
-1.6
30
35
40
45
50
-10
-5
0
5
10
15
20
25
30

Experimental Setting

- Baseline system: phrase-based extension of IBM Model-4 (6 feature functions)
  - Beam-search decoder: threshold pruning, histogram pruning
- Task 1: Chinese-English NIST 2003 Large data condition
  - domain: news agencies
  - statistics: vocab: CH 148K, EN 110K; #words: CH 13.1M. EN 13.5M
  - develop/test: 877sp with 4 references/919sp with 4 references
- Task 2: Chinese-English C-STAR Eval 2003
  - domain: basic traveling expressions
  - statistics: vocab: CH 12K, EN 11K; #words: CH 434K, EN 450K
  - develop/test: 1,000sp with 1 reference/506sp with 16 references
- Translation Error Measures: BLEU/NIST scores
- Performance Measures: BLEU/NIST scores versus search complexity (#hyp)

Results

Additional information:

- Baseline uses uniform parameters
- Beam-search settings:
  - loose threshold-pruning
  - tight histogram pruning
- Minimum Error Training:
  - with 12 CPUs - Xeon 2.4GHz
  - single iteration takes 7min
  - convergence in about 100 steps

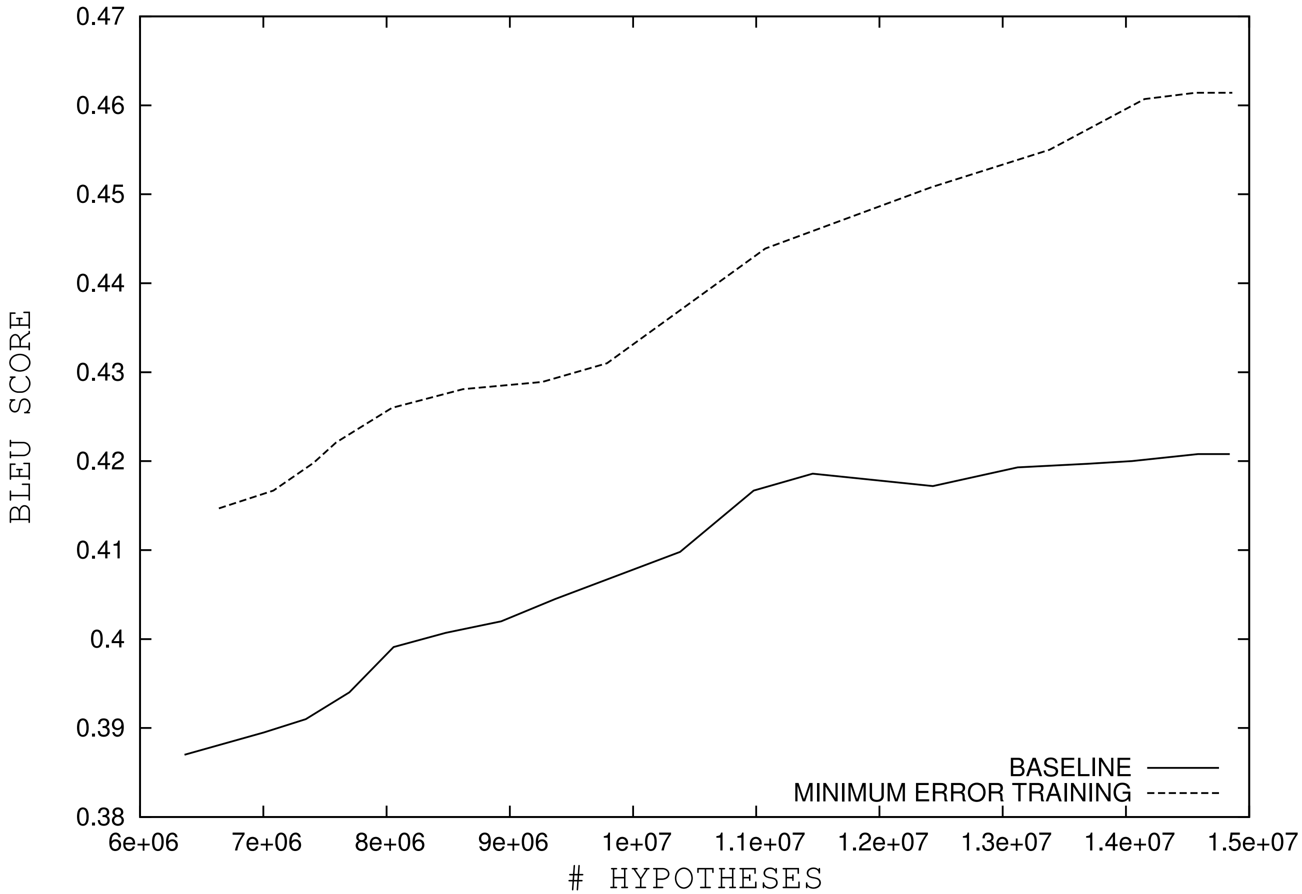
NIST 2003 TASK

Criterion	BLEU	NIST	# hyp
BLEU	0.1854	7.2882	116M
NIST	0.1840	7.3362	115M
Baseline	0.1803	7.2115	116M

C-STAR 2003 TASK

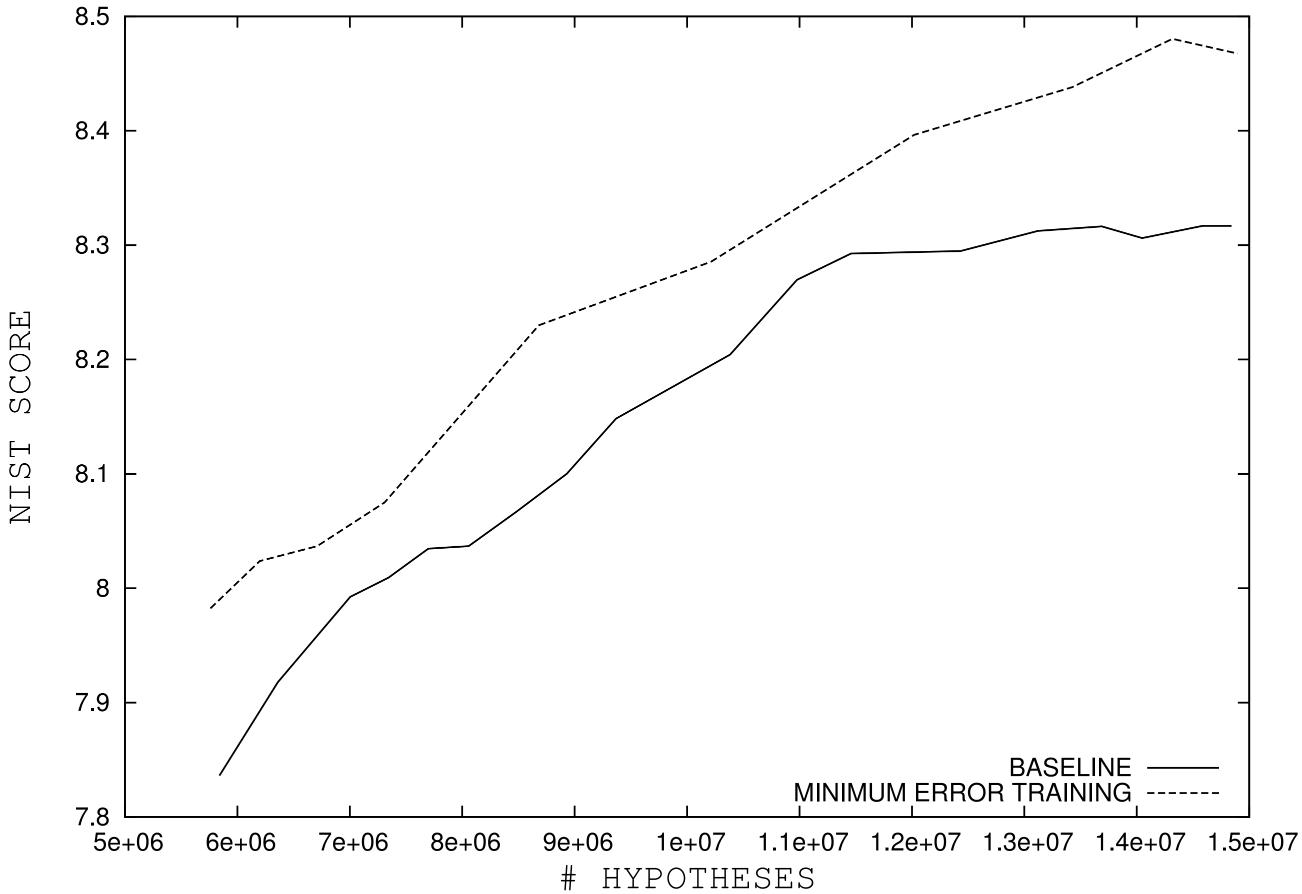
Criterion	BLEU	NIST	# hyp
BLEU	0.4614	8.4945	14.9M
NIST	0.4581	8.4675	14.9M
Baseline	0.4208	8.3169	14.8M

Performance after Parameter Training



BLEU score after applying threshold pruning.

Performance after Parameter Training



NIST score after applying threshold pruning.

Conclusions & Future Work

- Small but consistent and stable score improvements
  - however, no subjective assessment has been made
- BLEU score optimization is more effective than NIST score optimization
- Simplex can also be used to tune other parameters
  - e.g. pruning parameters of the search-algorithm [Zens & Ney, 2004]
- Future work will investigate:
  - the use of n-best lists and re-ranking methods [Shen et al., 2004]
  - the joint optimization of ASR and SMT model parameters [Zhang et al. 2004]