Применение LabelMover для оптимизации формы полюсного наконечника

Постановка задачи находится по адресу: <u>http://elcut.ru/advanced/pole_optimization.htm</u>

Шаг 1. Выбор типа задачи.

Запускаем LabelMover, он входит в поставку ELCUT, поэтому найти его можно в меню Пуск>Программы>TOR Coop>ELCUT>Tools, либо запустить прямо из ELCUT, нажав кнопку Параметрический анализ с помощью LabelMover.



LabelMover способен производить три вида расчетов: Последовательные расчеты, Статистический анализ и Оптимизацию.

В данном случае нам необходимо выбрать оптимизацию. Для этого жмем соответствующую кнопку на закладке **Начало**, либо **Файл**>**Создать**>**Оптимизация**

🗓 ELCUT LabelMover 2.2 - [Оптимизация1]	_ 🗆 🗙							
Файл Правка Вид Инструменты ?								
Начало Значения Параметры Результаты График	L							
Последовательные расчеты Статистический анализ Оптимизация	Задать цель/значения							
Вы работаете с подсистемой оптимизации.	Задать параметры							
	Получить результаты							
Для знакомства с ее возможностями:	Шагов <= 1200 ▼							
• Нажмите Исходная задача и выберите задачу, которую вы хотите исследовать.								
 Нажмите Задать цель / знач и задаите значение, которое вы хотите оптимизировать. 								
• Нажмите Задать параметры и определите пространство поиска. В качестве								
параметра вы можете задать любое возможное изменение геометрии или физических свойств.								
• Нажмите Получить результаты, чтобы начать поиск оптимального варианта.								
F1 - дополнительная информация.								
Исходная задача: Исходная задача								
Для получения справки нажмите клавишу F1	1.							

Шаг 2. Выбор исходной задачи.

Нажимаем кнопку **Исходная задача** и выбираем файл задачи, в нашем случае это файл *pole optimisation.pbm*

(Если LabelMover был запущен из ELCUT, в котором наша задача уже была открыта, то она автоматически установится в качестве исходной).

При этом файл откроется в ELCUT:

🛃 Elcut - [Pole_optimization.mod]										
🛃 Файл Правка Вид Задача Сервис Окна ?										
🗋 🗅 🚅 🔚 🖳 🚑 🗢 🐇 🖻 💼 🎒 🤣 🦹 🙌 🔹 🔛 👫										
роle_optimization.pbm - нелинейна роle_optimization.r Физические свойства: Pole_optimization.r Физические свойства: Pole_optimization.r Справочник свойств: <нет> Связи задач: нет связей										
			•							
Для вывода справки нажмите клавишу F1		0.200 м, 0.340 м	1							

Шаг 3. Задание цели оптимизации.

Для начала зададим цель. В нашем случае это – максимальная индукция поля в области замера.

Нажимаем на кнопку Задать цель/значения. Программа открывает окно Добавление значений.



Слева видим метки блоков, ребер и вершин из задачи. Выбираем **Область**. Затем указываем величину, которую будем измерять. Для нас это **Средняя индукция по объему**. После этого задаем цель, которую необходимо достичь. В данном случае **Максимум**.

Нажимаем Добавить.

(Если необходимо параллельно вести измерения какой-либо другой величины, но не учитывать ее при оптимизации, то необходимо выбрать эту величину, а в поле Цель установить Не цель.)

Когда всё выбрали, жмем Закрыть и видим цель (либо цель и список дополнительно-выводимых значений):

ſ	🕦 ELCUT LabelMover 2.2 - [Оптимизация1]								
9	Файл Правка Вид Инструменты ?								
[
Γ	Начало		Значения	Парамет	тры	Результаты	График		
	Имя Цель								
	🗼 Средняя индукция по объему для область Максимум								

Шаг 4. Задание изменяемых величин.

Нажимаем на кнопку Задать параметры. Появляется окно Добавление параметров оптимизации. Для начала установим тип - Изменение геометрии, поскольку мы будем менять форму наконечника. Метод изменения – Перенос, так как мы будем двигать заранее подготовленные вершины.

Далее, выбирая в списке слева вершины, устанавливаем для них вектор смещения. Самые крайние вершины (*Bepx_1,2*) будем двигать только вниз. Средние (*Cepeduna_**) в обе стороны, а нижние (*Hus_1-6*), соответственно, только вверх:



То есть, например, вершину *Середина_3_1* можно двигать только на 5 мм вниз и на 15 мм вверх. Поэтому для dy1 задаем значение -0.005м, а для dy2 0.015м. По оси x движения не происходит, оставляем в полях нули.





Задав перемещения для всех вершин, жмем Закрыть и видим весь список параметров:

I	🕦 ELCUT LabelMover 2.2 - [Оптимизация1]							
	Файл Правка Вид Инстр	ументы <u>?</u>						
	D 🚅 🖬 🖻 🦹 👘							
	Начало Знач	ения	Параметры	Результаты	График			
I	Описание			Диапазо	н			
I	/ Переместить верх_1			(0, -0.02) (0, 0)			
I	/ Переместить верх_2			(0, -0.02) (0, 0)			
I	/ Переместить низ_1			(0, 0) (0, 0.02)			
I	/ Переместить низ_2			(0, 0) (0, 0.02)			
I	/ Переместить низ_3		(0, 0) (0, 0.02)					
I	/ Переместить низ_4			(0, 0) (0, 0.02)			
I	/ Переместить низ_5			(0, 0) (0, 0.02)			
I	/ Переместить низ_6			(0, 0) (0, 0.02)			
I	🖊 Переместить середина_1	_1	(0, -	0.015) (0,0.005)			
I	🖊 Переместить середина_1	_2	(0, -	0.015) (0, 0.005)			
I	🖊 Переместить середина_2	2_1	(0	, -0.01) (0, 0.01)			
I	🖊 Переместить середина_2	, -0.01) (0, 0.01)					
Переместить середина_3_1 (0, -0.005) (0, 0.015)								
	🖊 Переместить середина_3	_2	(0, -	0.005) (0,0.015)			
П	1							

Шаг 5. Расчет и получение результатов.

Сохраним наш файл оптимизации перед окончательным запуском расчета. Файл>Сохранить как.

Теперь необходимо выбрать число шагов. Для подобной задачи их должно быть несколько сотен, чтобы программа «успела» подвигать все точки. Однако, чем больше число шагов, тем дольше длится расчет.

Например, при получении результата на странице примера (см. ссылку в начале статьи), было установлено 600 шагов.

В данном примере установлю 100 шагов в целях демонстрации. Точность результата при этом будет ниже.

🛍 ELCUT LabelMover 2.2 - [pole_optimisation.qva]									
Файл Правка Вид Инструменты ?									
🗅 🚅 🔚 🐚 🤶									
Начало Значения	Параметры	Результаты	График	<u> </u>					
Описание		Диапазон		Задать цель/значения					
Переместить верх_1	(0, -0.02) (0, 0) (0, -0.02) (0, 0)			Задать параметры					
Переместить низ_1		(0, 0) (0, 0.02)	2	Получить результаты					
Переместить низ_2		(0, 0) (0, 0.02)	1	<u>Ш</u> агов <= 100 💌					
Ш / Переместить низ 3		(0 0) (0 0 02)							

Нажимаем Получить результаты.

Программа начнет генерировать и решать задачи. За ходом решения можно следить в появившемся окне, а также наблюдать открывающиеся в ELCUT файлы.

🛄 ELCUT LabelMover 2.2 - [pole_optimisation.qva]Хайл Правка Вид Инструменты ?									
ŀ	Начало		Значения	Параме	етры Ре	зультаты	График		1
Шаг	Пере	мес	Перемес	Перемес	Перемес	Перемес	Перемес	Пе	Задать цель/значения
0	0, -0	0, 0 .002	0, 0 0, 0	0, 0 0, 0	0, 0 0, 0	0, 0 0, 0	0, 0 0, 0		Задать <u>п</u> араметры
2		0, 0 0. 0	0, -0.001 0, 0	0, 0 0. 0.001	0, 0 0, 0	0, 0 0, 0	0, 0 0, 0		Получить результаты
4		0, 0	0, 0 0, 0	0, 0	0, 0.00239	0, 0	0, 0 0, 0		Шагов <= 100 💌
6		0,0	0,0	0,0	0,0	0,0	0, 0.0011	0.	Посмотреть модель
8		0,0	0,0	0,0	0,0	0,0	0,0	-,	Картина <u>п</u> оля
10		0,0	0,0	0,0	0,0	0,0	0,0		<u>К</u> онтуры
12		0,0	0,0	0,0	0,0	0,0	0,0		
13 14		енерир	уются и реш	аются зада	чи [17%]				
16 0, 17 0, Внешняя итерация: 8 Построение и факторизация матрицы									
Отмена									

Попутно программа заносит в список результаты, а также сохраняет каждую из сгенерированных задач в папку *pole_optimization_QLM_Files*, которая находится в директории исходной задачи.

После завершения программа отметит в списке требуемый результат. У нас это максимальная индукция в области замера:

🗓 ELCUT LabelMover 2.2 - [pole_optimisation.qva]									
Файл Правка Вид Инструменты ?									
	Начало		l						
ec	Перемес	Перемес	Перемес	Перемест	Перемес	Средняя		Задать цель/значения	
000	0, 0.000	0, 7.682	0, 0.000	0, 0.0031	0, 0.000	0.67772		2	
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.68189		Задать параметры	
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.6811		Получить результаты	
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.6824			
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.67421		Шагов <= 100 ▼	
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.66148			
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.65569		Посмотреть модель	
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.67182			
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.68242		Картина <u>п</u> оля	
þ2	0, 0	0, 0	0, 0	0, 0.002924	0, 0.004	0.68351		I Course of Cour	
0,0	0, -0.006	0, 0	0, 0	0, 0.002924	0, 0.004	0.67872		Контуры	
0, 0	0, 0	0, 0.002	0, 0	0, 0.002924	0, 0.004	0.68356			
0,0	0, 0	0, 0	0, -0.005	0, 0.002924	0, 0.004	0.67929			
0,0	0, 0	0, 0	0, 0	0, 0.0004	0, 0.004	0.68185			
0,0	0, 0	0, 0	0, 0	0, 0.002924	0, 0.000	0.6815			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.0025	0, 0.003	0.64575			
41	0, -0.000	0, 9.058	0, -0.000	0, 0.0028	0, 0.004	0.66433			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.002562	0, 0.003	0.65277			
71	0, -0.000	0, 9.381	0, -0.000	0, 0.0028	0, 0.004	0.66852			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.002649	0, 0.003	0.66133			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.0027	0, 0.004	0.67088			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.0026	0, 0.003	0.66566			
þo	0, -0.000	0, 0.000	0, -0.000	0, 0.0027	0, 0.004	0.67344	-		
						►			
								11.	

Двойной клик по строке результата откроет соответствующую задачу в ELCUT, где мы можем увидеть искомую форму.



Результат отличается от представленного на странице примеров по причине низкого числа шагов. На закладке **График** мы можем посмотреть зависимость искомой величины от номера шага.



Чтобы убедиться в том, что оптимизация завершена, обычно устанавливают вдвое большее число шагов и запускают заново. Если конечные результаты не будут сильно различаться, то искомая форма найдена.

Пешехонов Д.

http://elcut.ru/