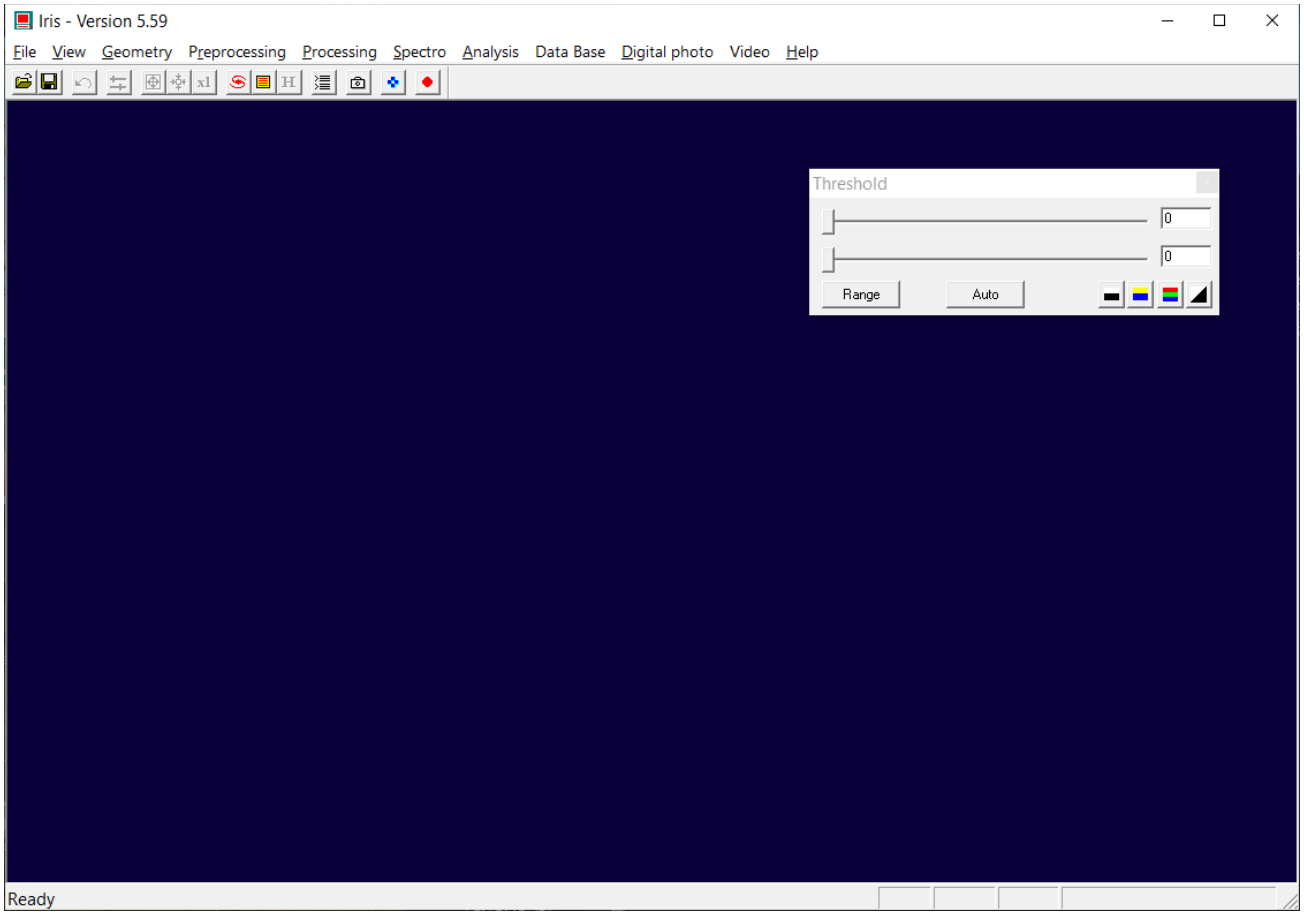
Обработка фотографий в Ирисе.

Этот документ я решил написать после того, как много раз после обработки чужих астрофотографий я получал вопросы «А в какой программе ты это сделал?» И «А как ты ТАМ это сделал?». Первым делом хочу сказать, что возможно это руководство ещё будет дополняться, изменяться и систематизироваться, т.к. появление его пока что весьма спонтанно и скорее здесь излагается некий кулинарный рецепт, нежели основы.

Я по умолчанию пока предполагаю, что читателю известно, как получать кадры, что такое процесс калибровки, что такое вычитание тока смещения, темнового тока и деления на плоское поле. А также зачем всё это надо.

Итак — небольшое руководство по использованию Ириса от меня, известного на астрофоруме как 4D, и как Антон Чечкин в миру.

После запуска программы мы видим вот такое окно:

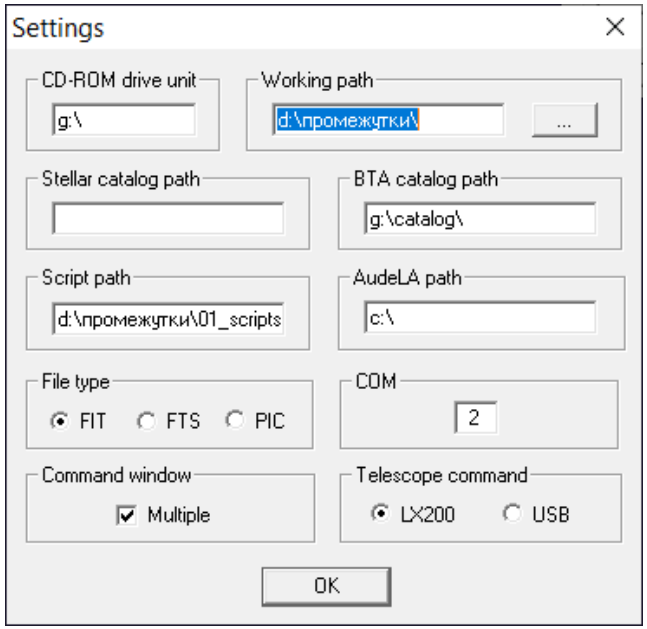


Первое, что мы должны сделать: задать рабочую папку, куда ирис будет сохранять свои файлы и тип этих самых файлов.

Идём в меню

File → Settings…

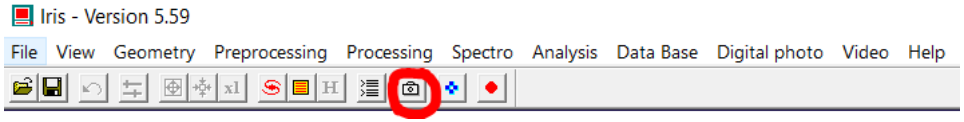
У меня всё это выглядит вот так:



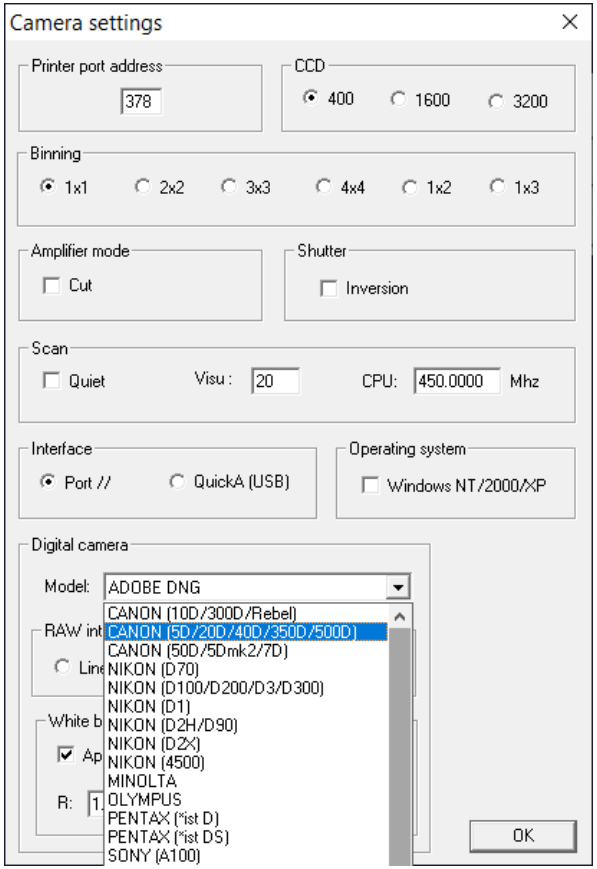
До недавнего (2008-2009 годы 😆) Ирис не поддерживал работу с цветными fit файлами. В то время как fit — это стандарт в мире астрономии. Однако уже довольно давно эту досадную оплошность исправили, так что ставим fit, и наслаждаемся. Выделенное поле как раз показывает рабочую папку. Также, поскольку ирис чрезвычайно настойчиво рекомендует нам работать из командной строки, он позволяет выполнять скрипты: файлы написанные вручную, как последовательность команд, и выполняемых также – последовательно. Как нетрудно догадаться за это отвечат поле с названием Script path. Если вы немного знакомы с программированием и вам нужно обработать, например серию кадров, в каком-нибудь нестандартном режиме — скрипты тут вам могут очень помочь. Неважно КАК вы их создадите – это всего лишь последовательность команд, записанных в текстовый файл с расширением “.pgm”.

Но об этом как-нибудь в другой раз, если у меня будет избыток свободного времени, а эта часть хоть кому-то пригодится. 🙄

Перво-наперво нам надо научиться загрузить файл. Если мы снимали на зеркальный фотоаппарат в формате RAW (мы ведь астрофотографы-профессионалы, всё важное мы снимаем только в RAW?! 🙄), то нам надо кликнуть по значку фотоаппарата:



И в открывшемся меню надо выбрать свою (или хотя бы максимально похожую) модель камеры.

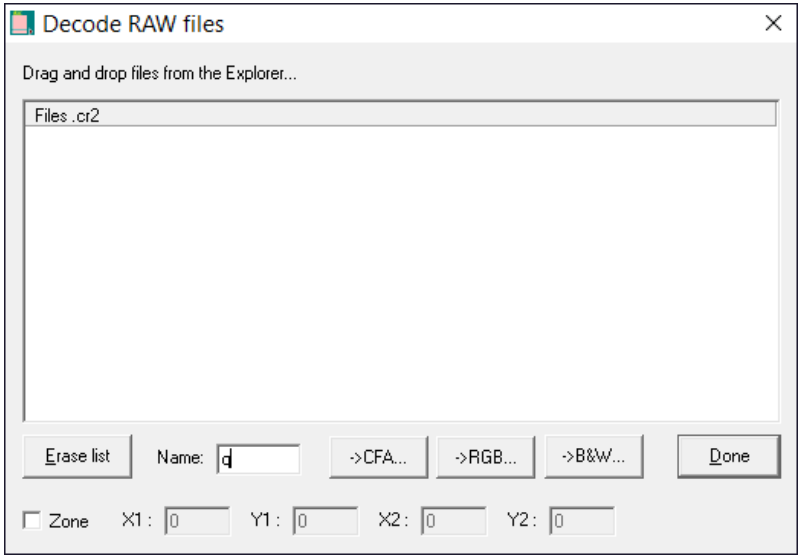


Все остальные параметры типа биннинга, значения RGB баланса и прочего, в данном окне влияют хоть как-то только если вы управляете камерой прямо из ириса, что лично я не делал ни разу. Поэтому вряд ли вам что-то здесь нужно менять.

Если мы продолжаем обсуждать самое начало работы, а именно – загрузку RAW файлов в Ирис, то дальше мы выходим из этого окна, и идём в меню:

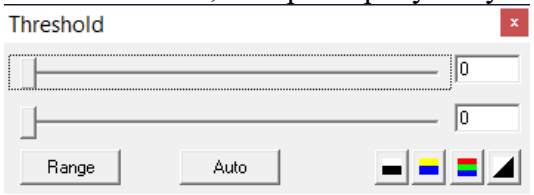
Digital Photo → Decode RAW files…

Что вызывает сворачивание окна программы и открывает окно загрузки (которое может оказаться под другими, уже открытыми окнами Windows, да, вот такой баг):



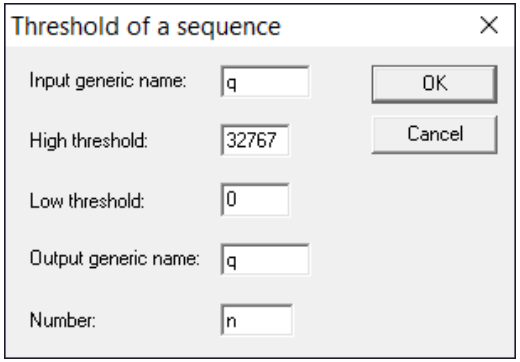
Сюда мы можем методом Drag and drop (Перетаскивания) закинуть те файлы, которые хотим конвертировать, в поле “Name” задать им префикс, жмякнуть на -> CFA (т.к. калибровку правильно осуществлять ДО дебайеризации) и получить серию изображений вида:

q1, q2, … qn с расширением «.fit», где n — число загруженных нами кадров. После этого нажимаем Done, и возвращаемся в программу, которая уже сама выставит уровни в окошке Threshold согласно тем значениям, которые присутствуют в файле:



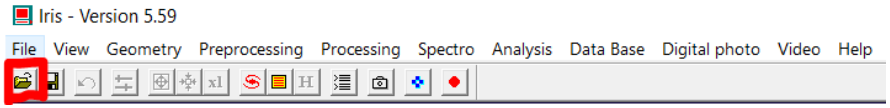
Подсказка. Если вы эти ползунки ранее трогали сами, то все последующие загруженные файлы будут открываться при том положении ползунков, при котором они были загружены, а не при значениях, выставленных автоматически. Это конечно можно легко изменить позже. Например, из меню:

View → Threshold of a sequence



Имя выходной серии при этом необязательно должно совпадать со входной. Это зависит от того, нужны ли вам файлы с исходно выставленными уровнями. Да, я надеюсь, что вы уже и без меня знаете, что значения в окошке Threshold никак не влияют на исходный файл, а лишь на его отображение на экране. Всё что находится ниже нижнего уровня показывается чёрным, выше верхнего — белым, а то что в промежутке соответственно считается по правилу пропорций: чем ближе к максимальному значению, тем ярче. Сделано это из-за того, что диапазон яркостей пикселей с матрицы обычно слишком велик, а корректная установку баланса белого, фотометрия и прочие астрономические операции работают только на данных у которых яркость не была изменена нелинейным образом.

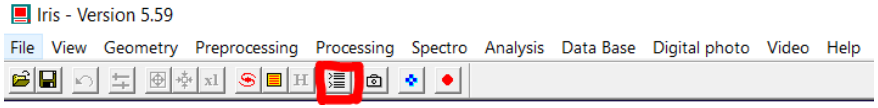
Однако я отвлёкся. После того, как мы загрузили изображения, они сохранились куда-то там на диск. И в программе отображается только последнее из серии. Как вывести например самое первое? Есть как минимум три способа:



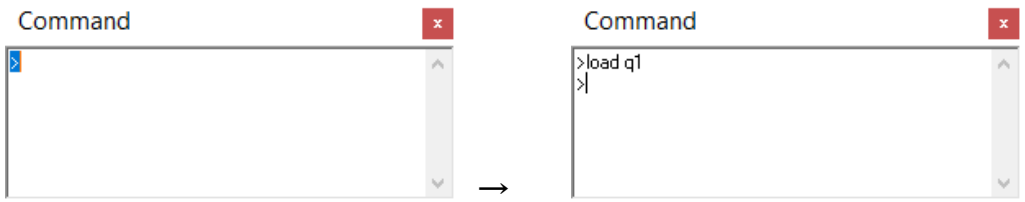
Из меню:

File → Load

И третий способ, на мой взгляд самый естественный для ириса, и самый непривычный для меня, как windows пользователя. Хотя за столько лет к интерфейсу ириса я уже вполне привык. Открыть уже-таки эту проклятую командную строку!



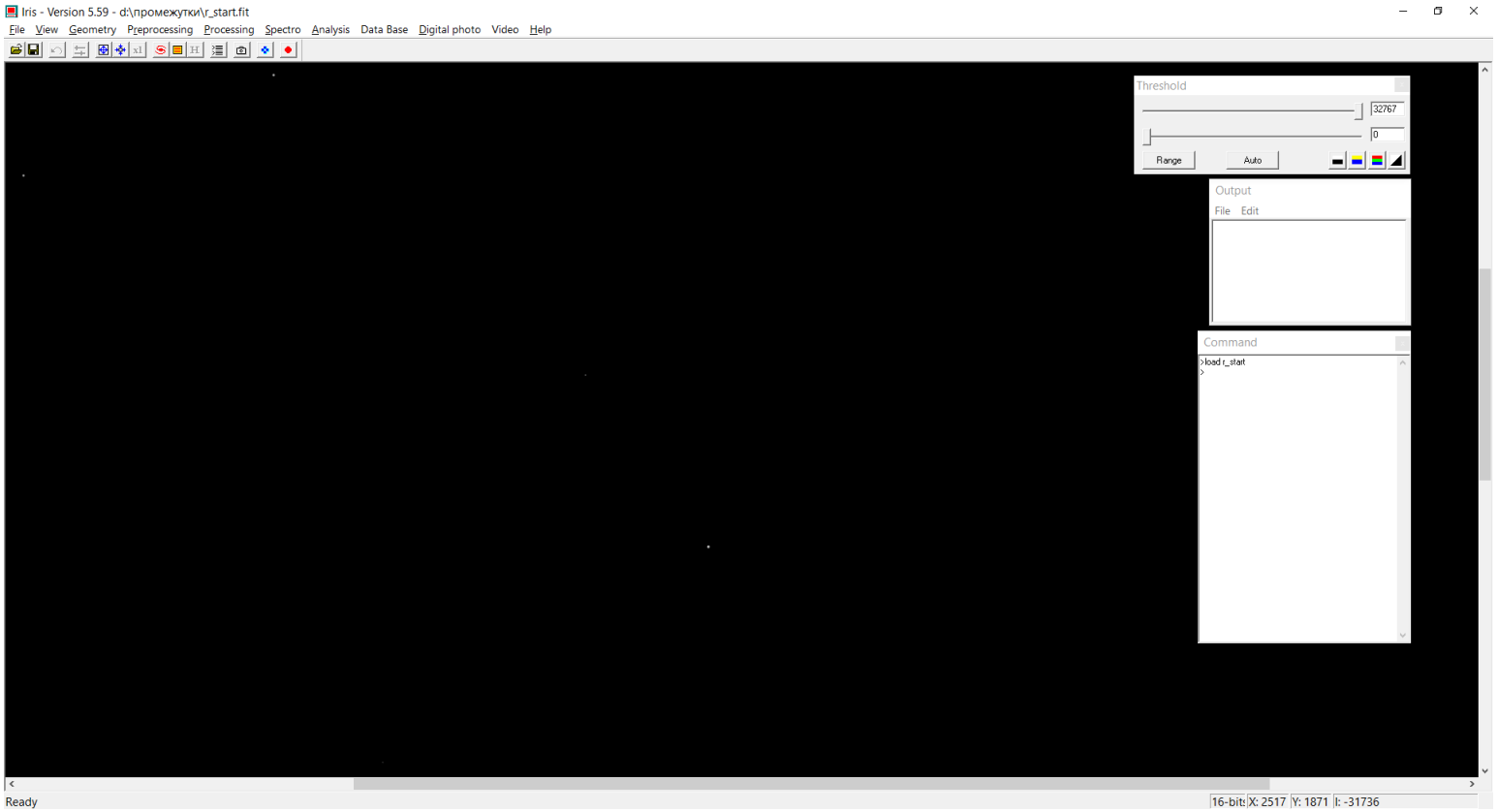
И увидеть вот такое окошко, которое я дальше по аналогии с операционными системами буду называть терминалом:



Где справа я с клавиатуры ввёл первую команду. Думаю, что тут перевод, и пояснения что это команда делает не требуется. 😊

Опуская пока что процессы калибровки, и выравнивания я хочу сосредоточиться на сборке LRGB изображений, где в качестве яркостного канала используется кадр, полученный без цветных светофильтров.

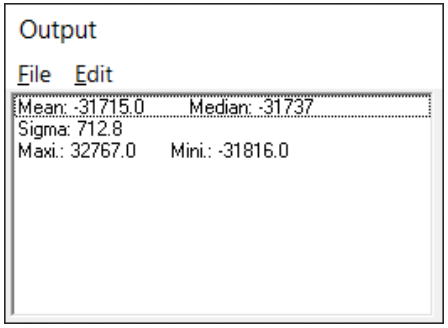
Предположим, что нам даны 4 уже готовых файла снятые на ЧБ матрицу, которые были уже заранее откалиброваны и приведены к единому положению. Назовём их R_start, G_start, B_start, L_start и положим в папку программы, которую указали ранее. Загрузим, например, кадр, соответствующий красному каналу и увидим «чёрный квадрат»:



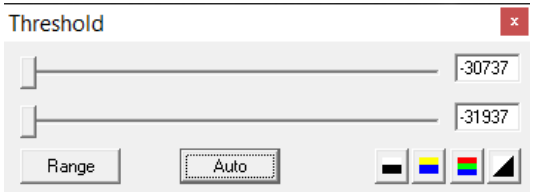
Если мы напишем в терминале команду

>stat

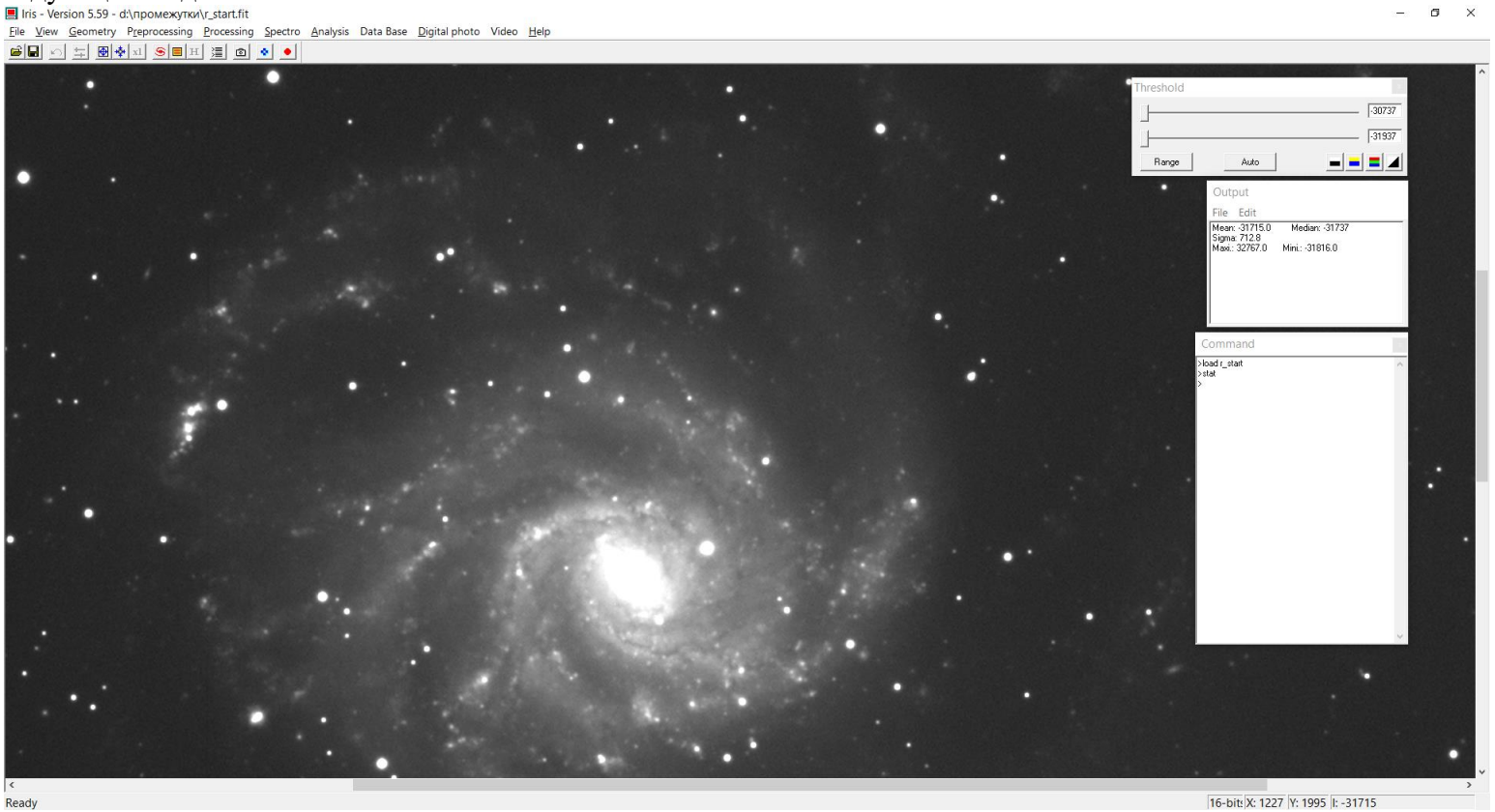
То вывалится окошко со статистикой:



Можно привести картинку в смотрибельный вид, кликнув по кнопке «auto» в окошке Threshold:



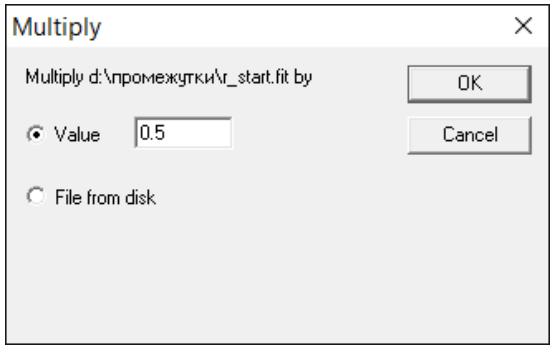
И получить следующий вид:

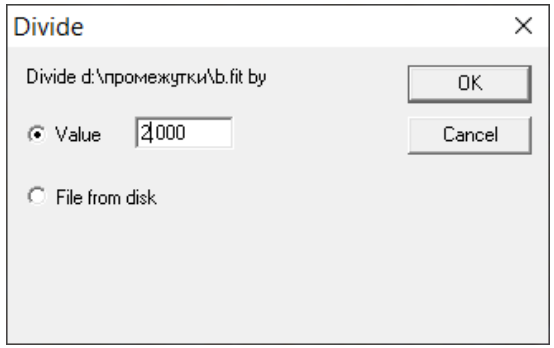


Можно видеть, что пиксели имеют значения практически от -32768 и до +32767, причём основная часть их лежит в отрицательной области. Ирис хотя и способен работать с отрицательными значениями, но нелинейные преобразования яркости на них выдают некорректные результаты. Поэтому для дальнейшей работы нам следует перевести всё в положительную полуось. Есть однако небольшая неприятность, рабочий диапазон Ириса строго ограничен во-первых, целыми числами, так что никаких 54,89 и прочих значений. А во-вторых — его диапазон также ограничен теми самыми -32768 — +32767. Для того чтобы уместить все значения пикселей нужно очевидно, поделить их пополам, что даст нам значения -16384 — + 16384. А потом ко всем пикселям добавить значение +16384. Так мы получим значения от 0 и до 32767.

Умножить значения на 0.5 можно опять аж целыми тремя способами. Первые два используют графический интерфейс, а последний — терминал:

Processing → Multiply… *ИЛИ* Processing → Divide…

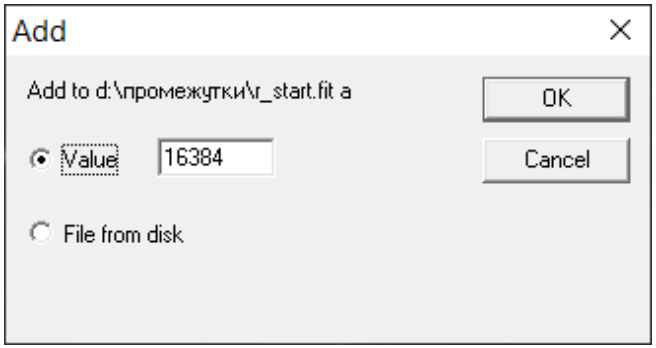


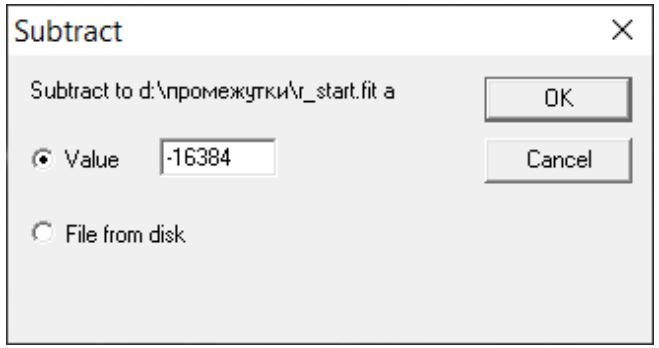


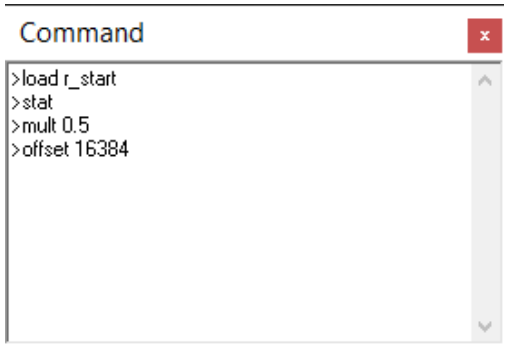
Надеюсь пояснения не требуются.

Дальше нам надо, как мы условились, добавить ко всем величинам константу. Опять три способа:

Processing → Add… ИЛИ Processing → Subtract…





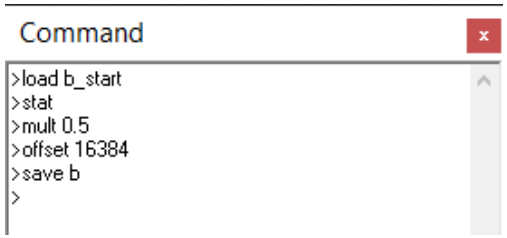


После чего сохраняем результат, вводя в терминал команду:

>save r

Хотя имя мы вольны выбирать любое, лишь бы без спец-символов и пробелов.

Далее, т.к. файлы у нас однотипные мы также можем обработать и их. При этом необязательно в терминале каждый раз вводить новую команду, можно также изменять уже имеющиеся, выставлять курсор на нужную строку и прожимать Enter. Вот так например выглядел мой терминал после обработки зелёного и синего каналов:



После того как мы подготовили ЧБ изображения — самое время собрать из них цвет и установить правильный баланс белого.

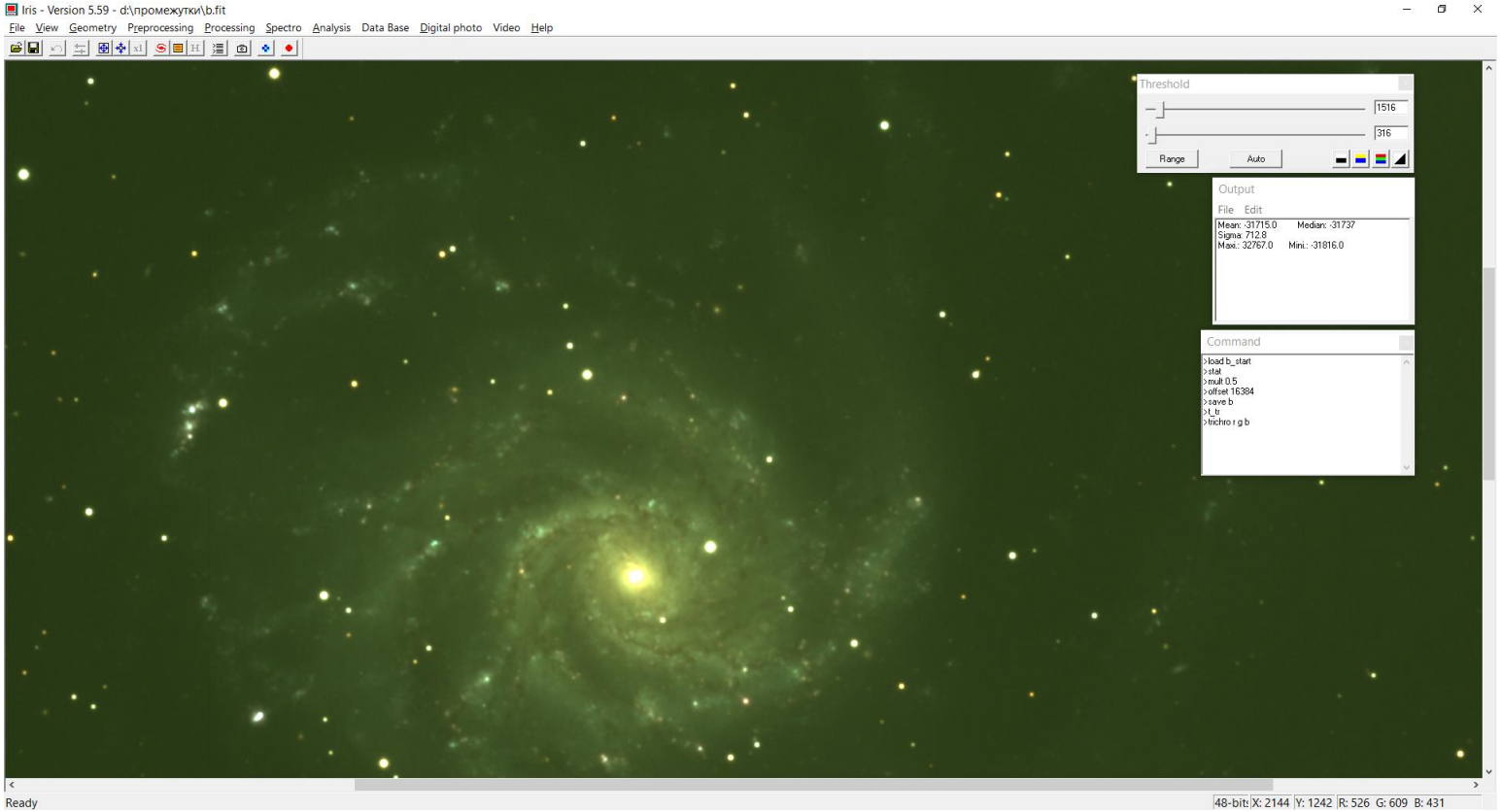
Для сборки цветного изображения из каналов есть две команды. Если имена каналов RGB r, g и b соответственно, то можно написать коротенькую команду:

>t_tr

Если же нет, то для этого есть команда:

>trichro [red] [green] [blue]

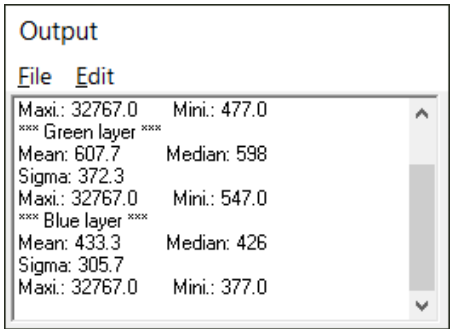
Где в квадратных скобках указаны имена соответствующих каналов. После применения любой из этих команд мы получаем вот такое изображение:



Где в окошке уровней ранее была нажата кнопка Auto. Уже интереснее, но баланс белого очевидно неправильный. Применив уже знакомую команду:

>stat

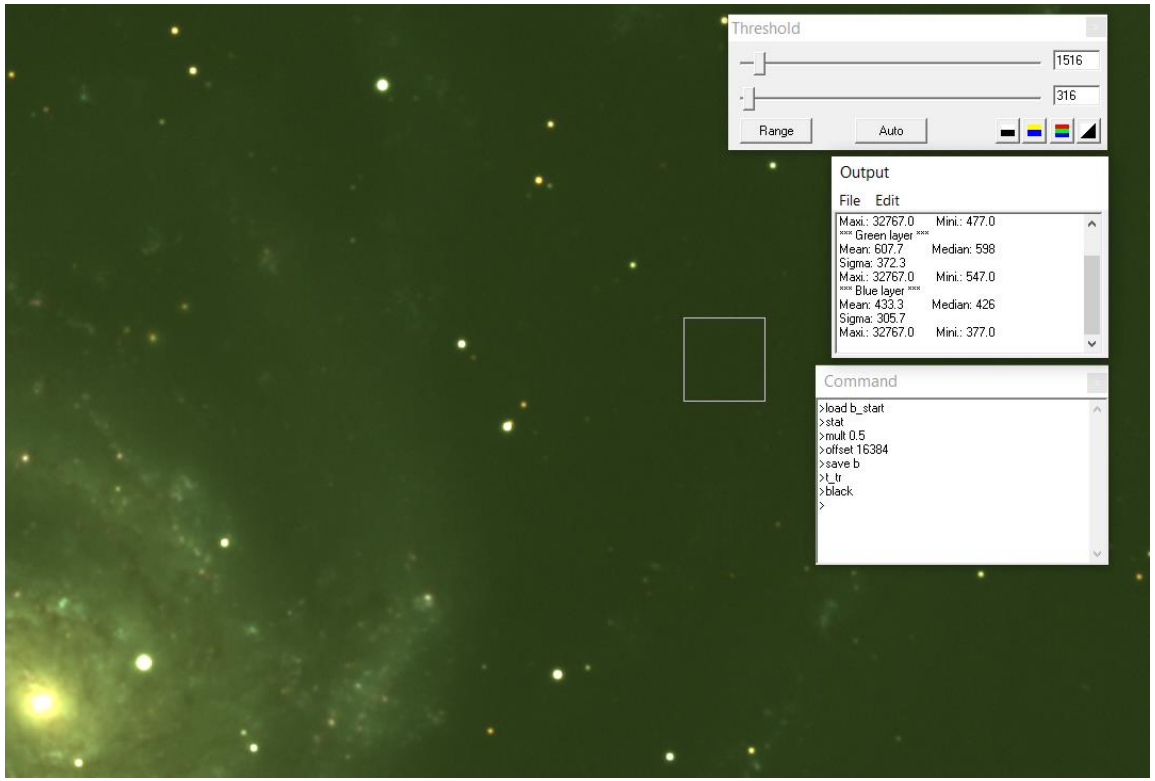
Мы видим следующую картину:



Алгоритм корректировки баланса следующий:

1) Мы должны привести фон к нейтрально серому, а лучше — чёрному цвету.
2) После нейтрализации фона необходимо растянуть (умножить) каждый канал на соответствующий множитель, чтобы картинка обрела человеческие цвета.

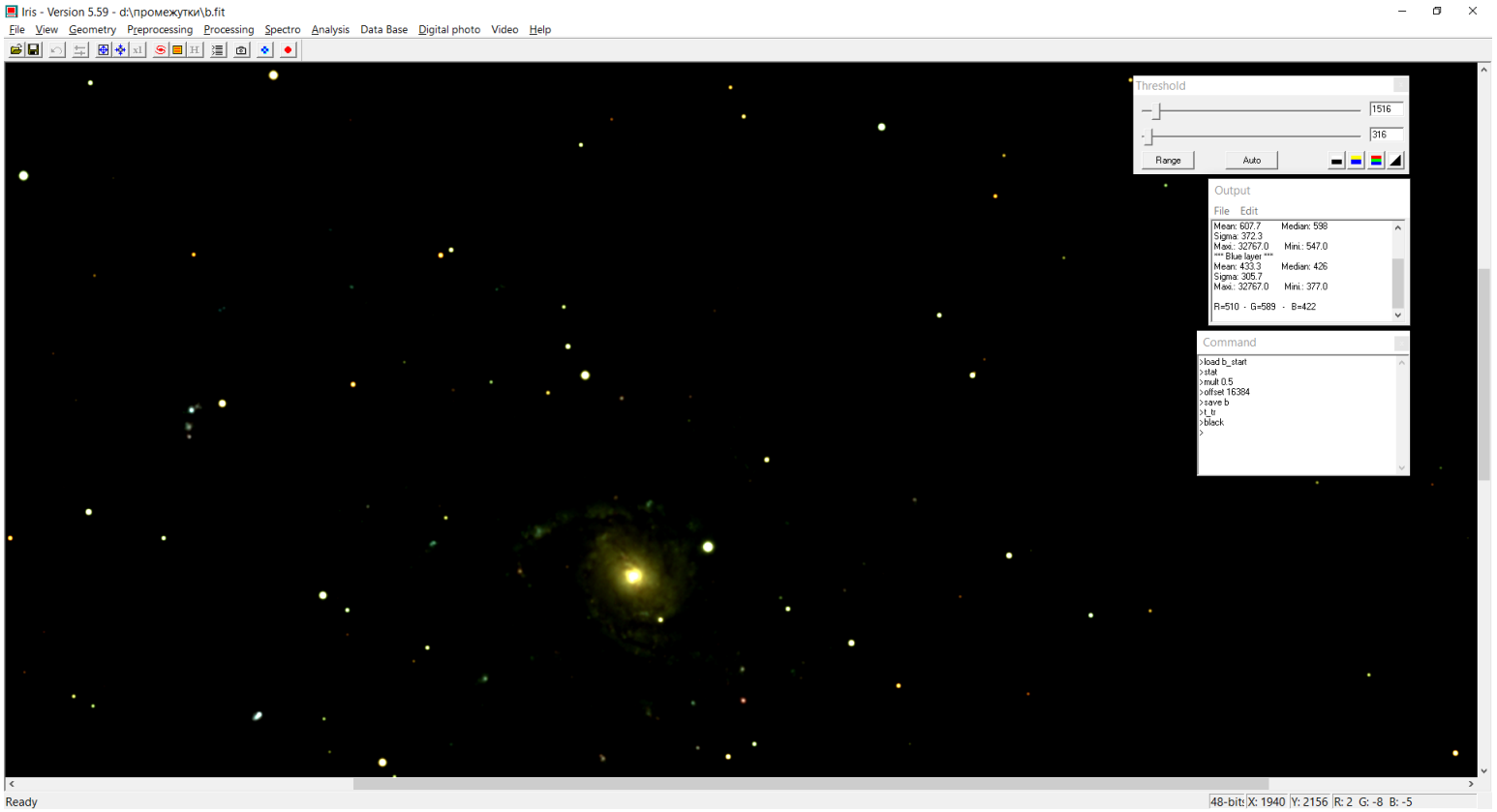
При этом понятно, что если уровень фона изначально не будет близким к нулю, то умножение его на коэффициент — будет его сдвигать, что усложняет процесс подбора правильных коэффициентов. Поэтому мы выделяем небольшой прямоугольник на области, где у нас нет звёзд и полезного сигнала:



И применяем в терминале команду:

>black

Получая:



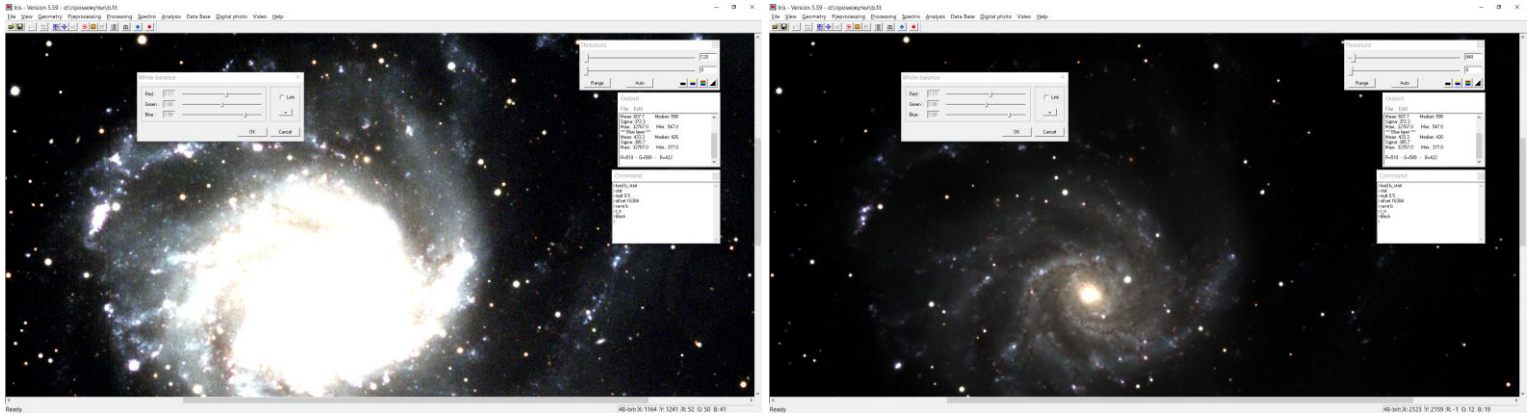
Здесь есть небольшое наблюдение. Лично у меня получается подобрать наиболее верный баланс, задрав уровни до предела. То есть мы оставляем ползунок чёрного там, где ему и положено быть — на нуле, а верхние уровни задираем до тех пор, пока не проявится структура галактики. После этого идём в меню:

View → White balance adjustment

И крутим ползунки до желаемого результата. Можно конечно ввести и из терминала:

>rgbbalance [coeff1] [coeff2] [coeff3]

Но это годится только если мы заранее знаем коэффициенты, т.к. подбирать таким способом — можно рехнуться, особенно с учётом того, что кнопки отмены в ирисе нет, и все промежуточные вычисления поэтому лучше записывать в новый файл.

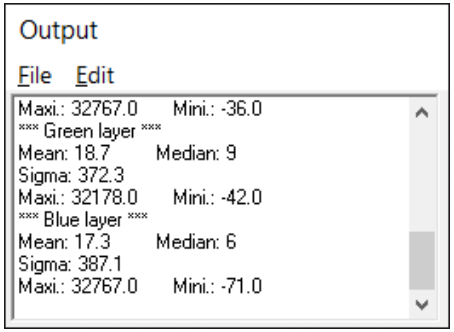


Подобрав подходящие коэффициенты и поиграв с уровнями для проверки, что остальные части кадра тоже выглядят естественно можно прожать OK.

После этого опять применяем нашу полезную команду

>stat

И видим в окошке минимальные значения в кадре:



Отрицательные. Нехорошо. Применяем, например

>offset 73

Также надо отметить, какие у нас максимальные значения. Если в кадре изначально была пересвеченная звезда, то максимальные значения во всех каналах ДОЛЖНЫ совпадать. Иначе потом, при наложении цвета мы получим некорректный результат. Если какой-то (обычно зелёный) канал «недожарен», то следует ВСЕ каналы умножить на величину $\frac{I_{max}}{I_{max}-I_{min}}$, посчитав эту величину на калькуляторе (или в уме 😳) и введя в терминал

>mult [coeff]

После всех манипуляций сохраняем результат, например как

>save color

Теперь пора разбираться с L-каналом. Если требуется, приводим его к диапазону 0-32767, как описано выше, а потом — один из самых естественных способов — прологарифмировать значения пикселей. При этом новые значения будут вычисляться по формуле:

$$y = k\frac{\lg x}{\lg x_{\max}}$$

Где x — исходное значение пикселя, $x_{\max}$ — пиксель на изображении с максимальной яркостью, а k — константа, на которую будет умножаться результат. Как обычно, эту команду можно ввести с терминала:

>log 32767

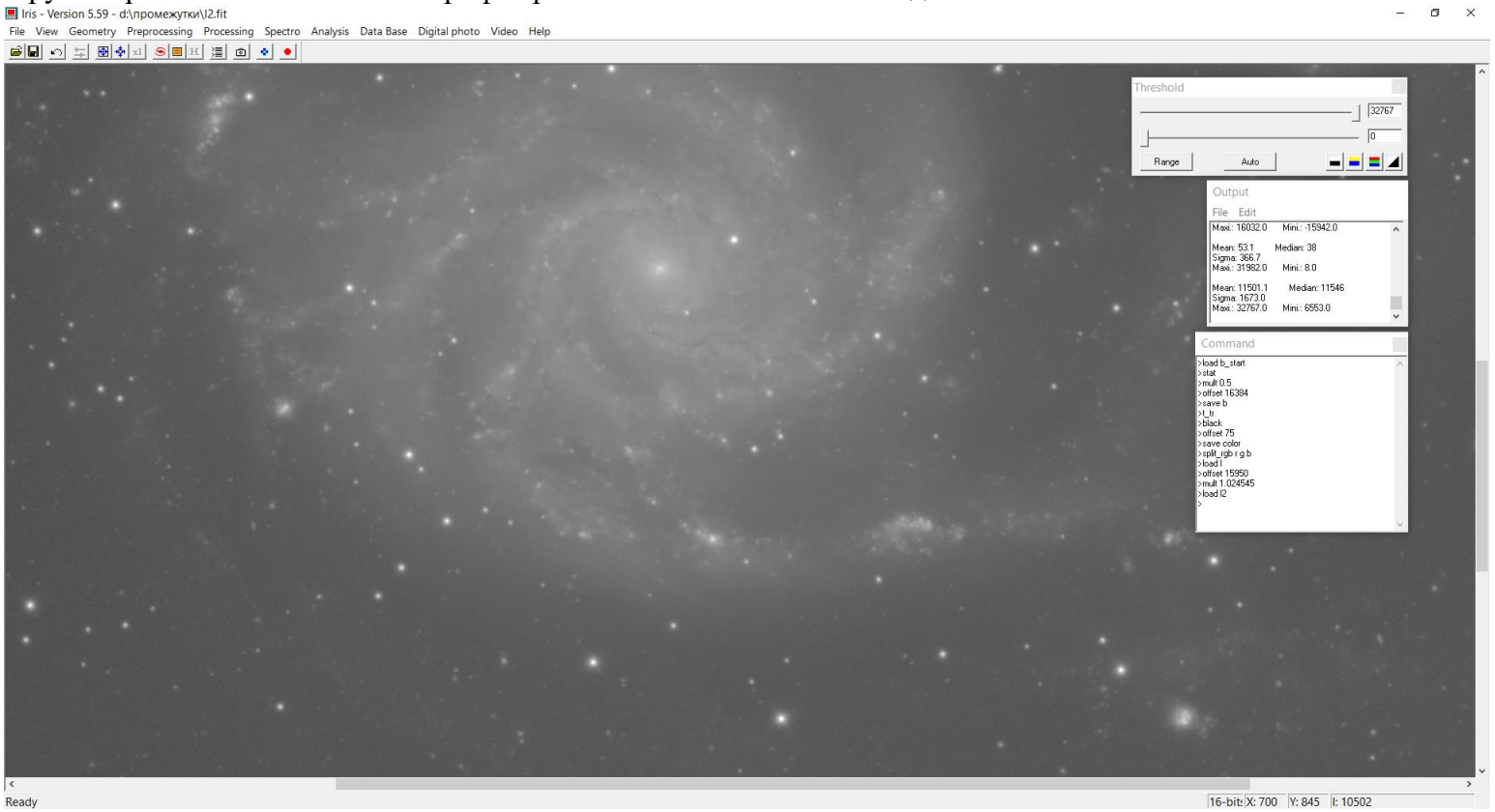
Или получить тот же самый результат из меню:

View → Logarithm

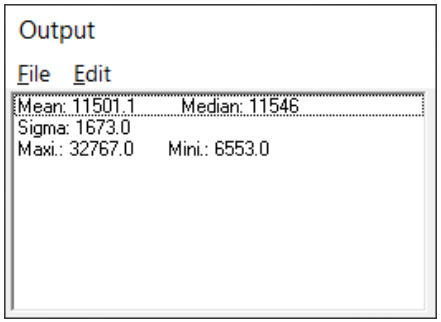
Но при этом значение $k = 32767$ будет подразумеваться по умолчанию.

Очередной небольшой лайвхак: если вы логарифмируете цветное изображение, то ирис делает это поканально. Т.е. если в каком-то канале нет пикселя с максимальной яркостью, то либо его нужно добавить вручную, либо баланс белого может значительно съехать (Для ЧБ изображений этой проблемы, очевидно, не существует).

Итак, после загрузки яркостного канала и логарифмирования мы имеем что-то подобное.



Посмотрим статистику, а она такова:



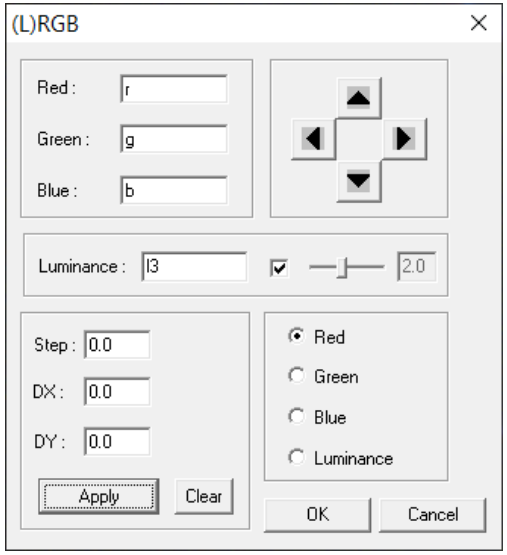
Чтобы использовать диапазон по полной применим:

>Offset -6550

>Mult 1.25

Где 1.25 взялось от $32767/(32767-6550)$. И сохраняем логарифмированную картинку, например как L3
А теперь попробуем сложить всё вместе, сходив в меню:

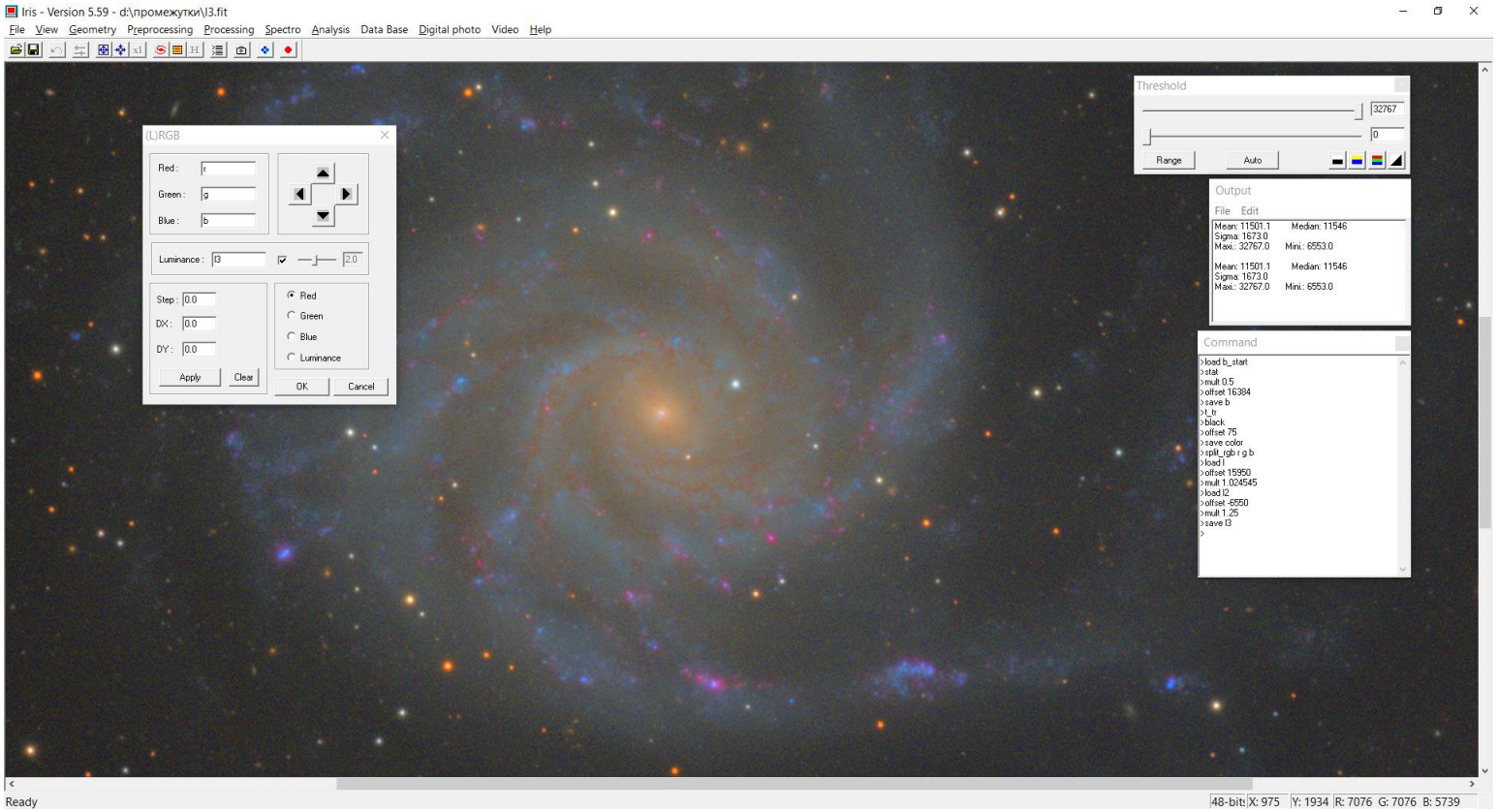
View → (L)RGB



Только вот ведь незадача, он требует от нас каждый канал по отдельности, а у нас цветной файл color.fit…
Ну да ничего. Мы загрузим наш color.fit и применим ещё одну команду:

>Split_rgb r g b

Которая, как нетрудно догадаться, разбивает цветное изображение на 3 отдельных файла с именами, которые мы можем задать сами. Теперь идём обратно и заполняем



Играя с ползунком luminance можно управлять насыщенностью цвета.

Этот стандартный метод в принципе неплох, но мне больше нравится пользовательский подход, т.к. тут слишком мало переменных, которыми мы способны осмысленно управлять. Поэтому я предлагаю ещё поразвлекаться с командной строкой.

У нас уже есть r g b файлы, которые мы можем истолковать иначе. Любой пиксель в цветном изображении характеризуется тремя числами. А со школы мы знаем, что в нашем трёхмерном пространстве любую точку можно тоже охарактеризовать тремя числами. Так мы пришли к идее цветового пространства. А как мы знаем, системы координат есть не только прямоугольные декартовы. Есть сферические, эллипсоидальные, цилиндрические, гиперболические… Почему бы нам пиксели не охарактеризовать какими-то другими величинами? Никто не запрещает. Есть, например, такая система как Lab, где у L отвечает за яркость пикселя, а две другие оси характеризуют его положение относительно «зелёный-пурпурный» и «жёлтый синий». Но мы сейчас рассмотрим другую систему, HSI, где три значения описывают Оттенок, Насыщенность и Яркость. Если мы теперь возьмём наше RGB изображение, перепишем с него величины, соответствующие оттенку и насыщенности, а яркость возьмём с логарифмированного кадра, то осуществим ту самую LRGB сборку. Но при этом мы, например, можем вмешаться в процесс на промежуточных этапах, что даёт нам больше свободы действий.

Для такого превращения RGB в HSI есть специальная команда:

```
>RGB2HSI r g b h s i
```

Очень «оригинально». Зато с запоминанием сложностей меньше. (Имена файлов по-прежнему можно задавать свои).

Если мы теперь введём туда наши исходники, загрузим изображение, ответственное за насыщенность и пару раз применим к нему команду

>gauss 0.7

>save s2

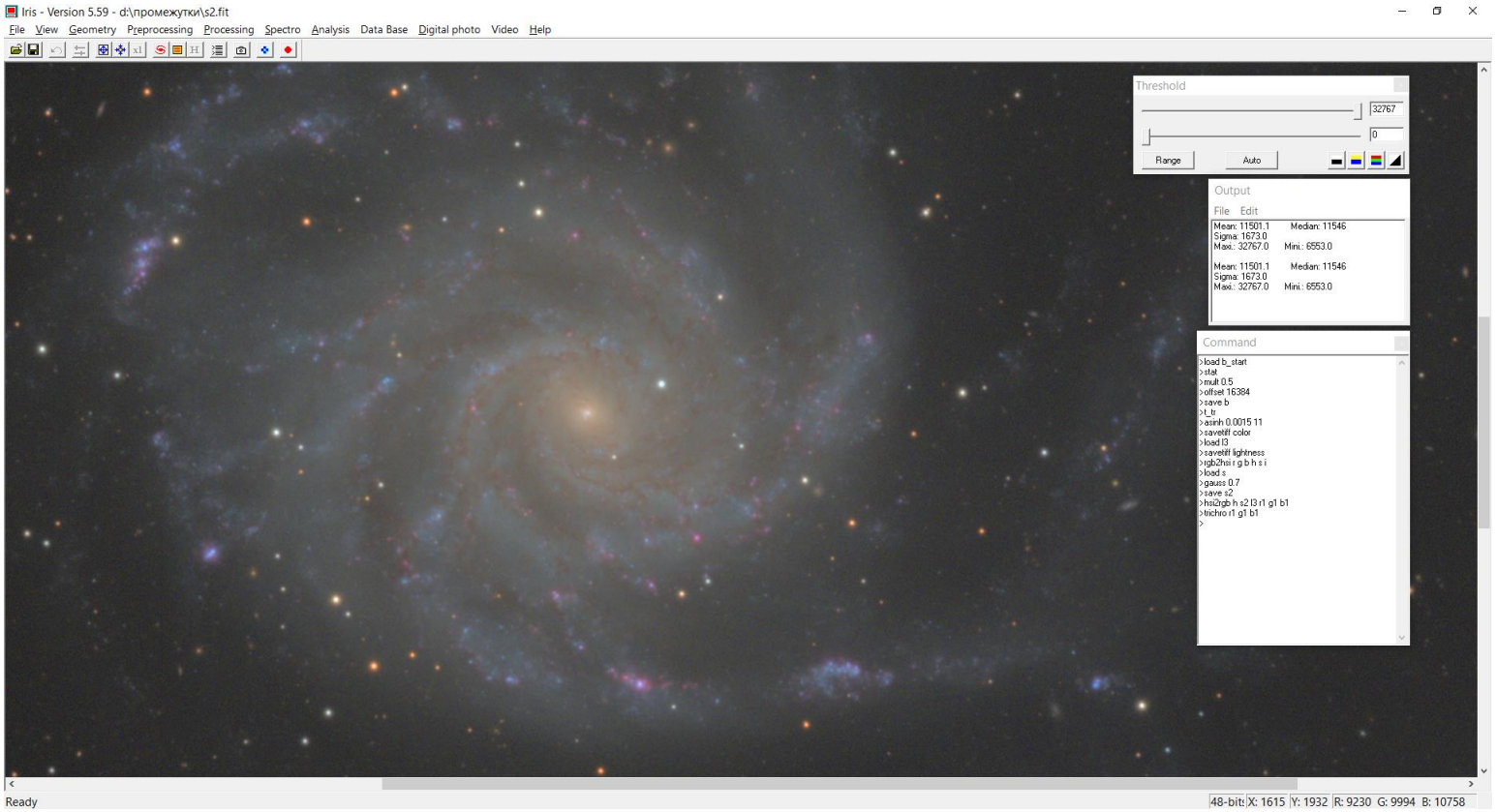
То мы уменьшим вклад от цветовых флуктуаций, и не исказим общую картину. Файл, ответственный за h размывать нельзя, т.к. иначе изменятся значения, ответственные за положения оттенков на цветовом круге, а нам этого не надо.

После того, как мы подготовили таким образом файлы h и s, можно провести обратное преобразование, думаю, что вы уже догадываетесь с какой командой:

>HSI2RGB h s2 L3 r1 g1 b1

Осталось только собрать изображение обратно:

>trichro r1 g1 b1



Теперь можно его сохранить и утащить, например в фотошоп.

>savetiff iris_lrgb

Как можно догадаться из названия команды, мы только что сохранили результат в tiff файл.

Казалось бы, что на этом можно закончить, но я всё-таки хочу показать, что хотя ирис и неплохой инструмент, но средствами фотошопа мы можем получить ещё более чистый результат.

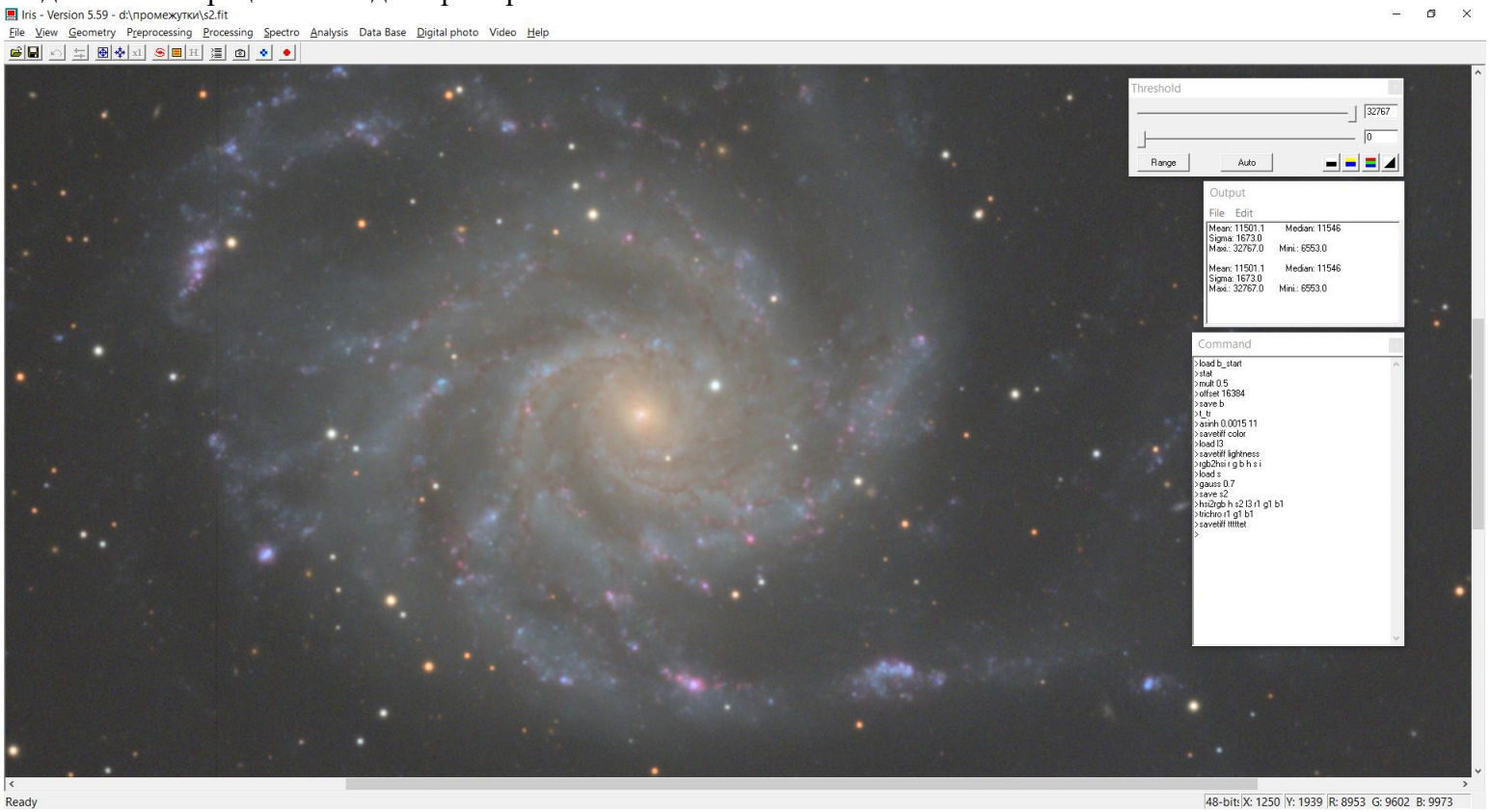
Загрузим наше цветное изображение и применим к нему дважды нелинейное растяжение гистограммы:

>load color

>asinh 0.0015 11

>asinh 0.0015 11

Результат выглядит этих операций выглядит примерно вот так:



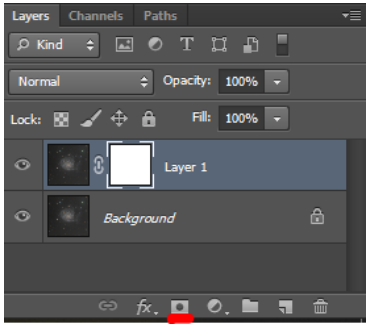
Дальше сохраняем его и наш преобразованный L-канал.

>savetiff color

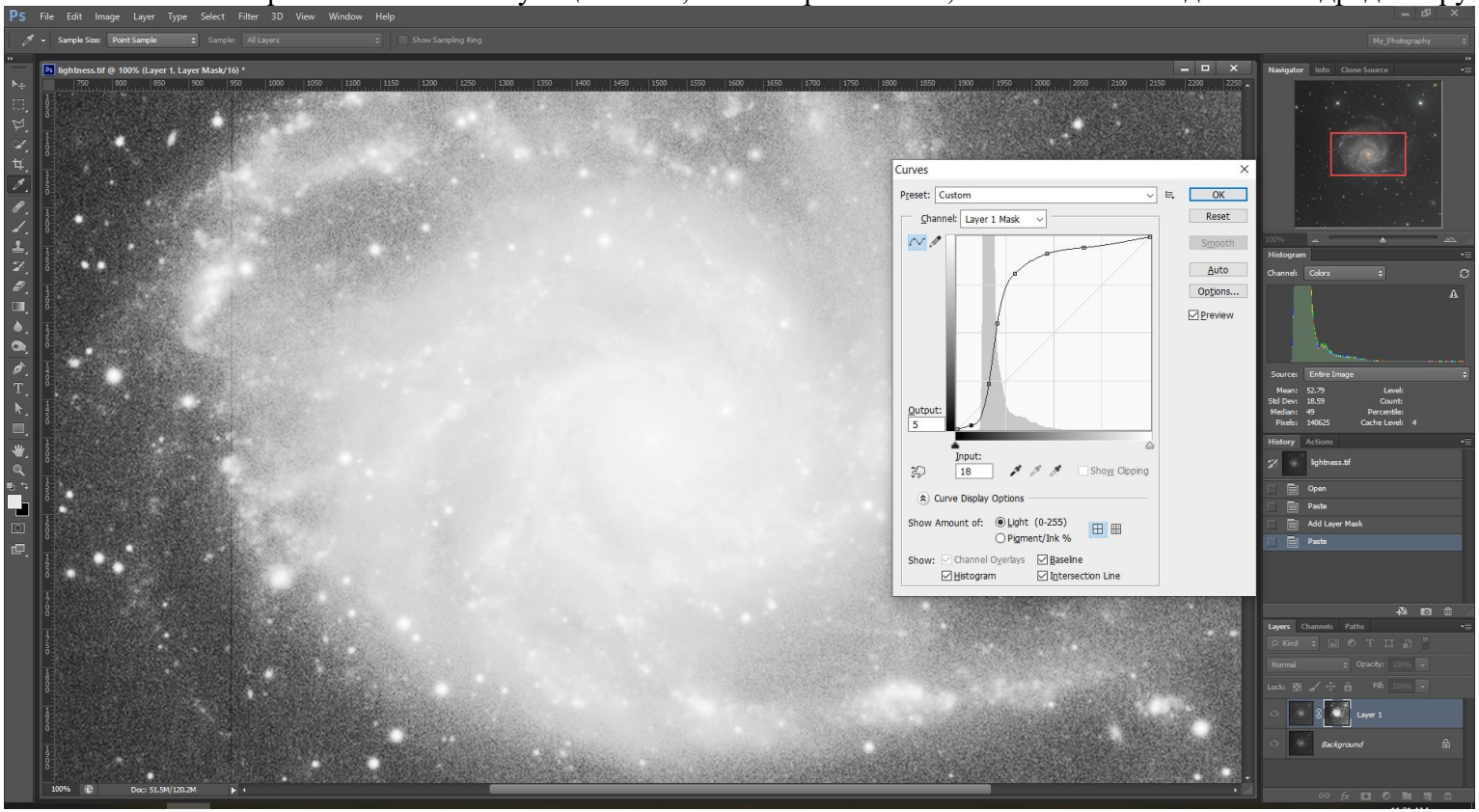
>load L3

>savetiff L

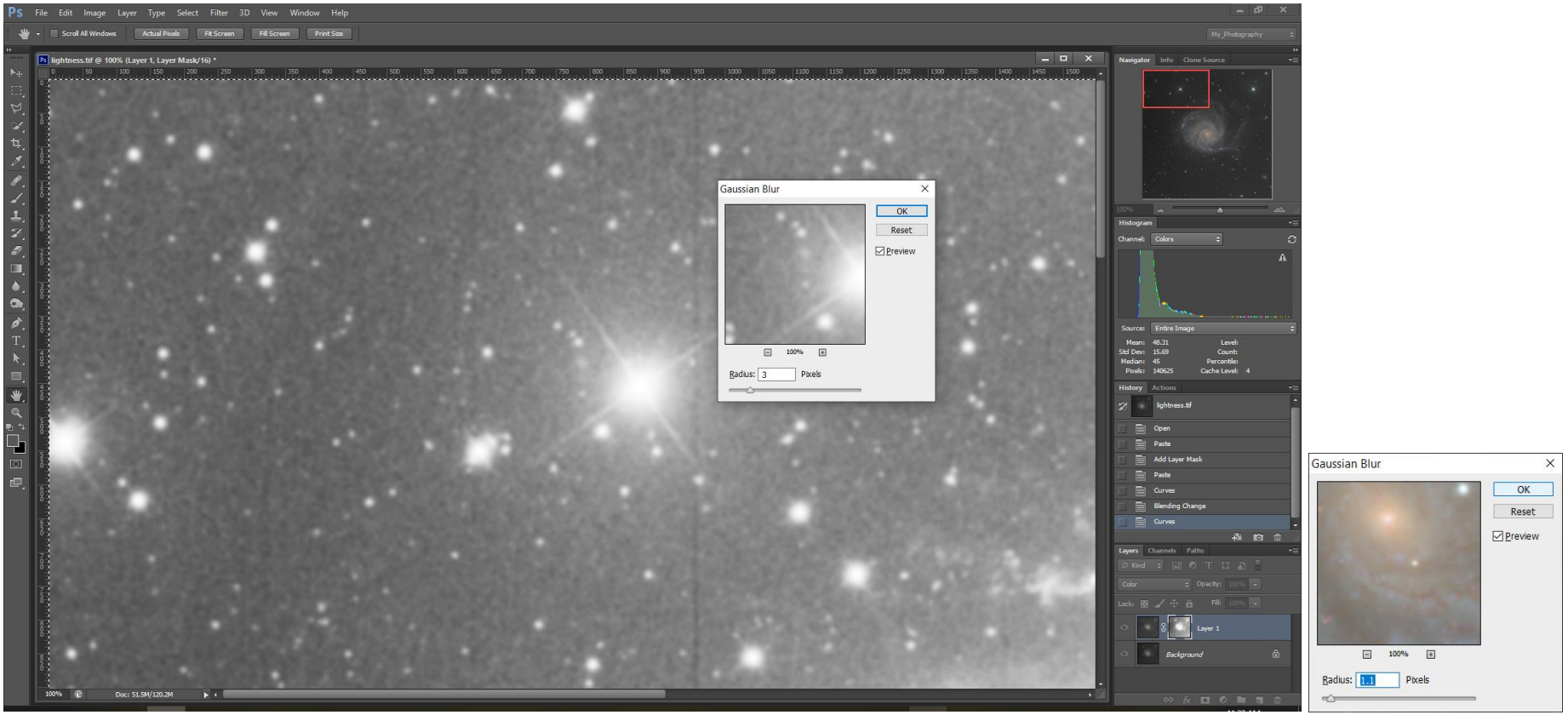
А теперь унесём наши тиффы в фотошоп и наложим цвет по маске яркости на изображение от L-канала.
Сначала скопируем то, что находится в яркостном слое, выделив его, а потом нажмём последовательно Ctrl+A и потом Ctrl+C.
Маску яркости можно создать нажав на иконку:



А для того чтобы войти в неё можно зажать клавишу Alt и щёлкнуть правой кнопкой мыши по появившемуся белому прямоугольнику. После этого вставляем скопированное изображение при помощи Ctrl+V.
Теперь белые области маски отображают соответствующий слой, а чем чернее маска, тем меньше вклад слоя. Подредактируем её кривыми:

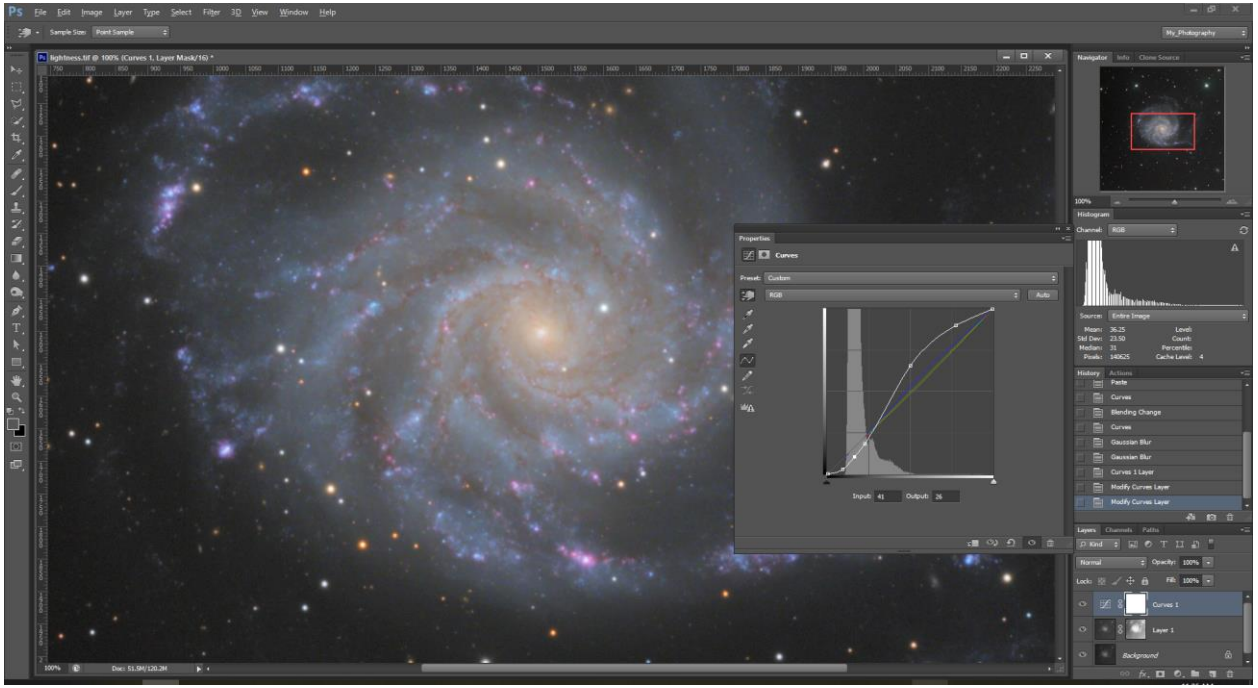


Далее следует поменять способ наложения слоя с цветом на «color» Таким образом яркие детали будут раскрашиваться, а тёмные участки небо оставаться практически нейтрально серыми. Также нам никто не мешает эту маску (как собственно и сам слой, содержащий цвет) немножко подразмыть гауссом, для более плавных границ и устранения цветовых артефактов.



Также, пока мы не сильно изменяли яркости и цвет, мы можем напрямую сравнить результаты полученные средствами одного только ириса и совместным применением Ириса и Фотошопа. Приведу увеличенные области с ядром галактики и периферией. Видно, что сложение по маске вносит гораздо меньше цветового шума в тёмные области изображения, так что оставим гибридную версию Ирис+Фотошоп для дальнейшей работы.

Собственно результат уже выглядит неплохо. Можно ещё немножко поднять контраст и исправить небольшое смещение баланса белого при помощи кривых. Помня при этом, что, да, кривые могут помочь при НЕБОЛЬШОМ избыточном оттенке. Но если баланс белого изначально был задан грубо неверно, то уже практически никакие попытки его восстановить ни к чему не приведут, так как преобразования цветов носят очень нелинейный характер и баланс необходимо задавать хотя бы в первом приближении верно ещё на линейном изображении!



На этом пока предлагаю закруглиться, а если появятся отзывы, дополнения, пожелания, то буду пробовать этот мануал как-то расширять и пытаться систематизировать, хотя пока – я слабо представляю во что всё это выльется.