

Lecture 9:

States of the Factor, Beam, and Grating

To accompany <https://youtu.be/Qb9zL0LztJo>

Version 01, released September 28, 2022

Paul Mirsky

paulmirsky633@gmail.com

1 Singular factors / Position factors

Diagram	State vector	Position value
	$(1, 0, 0, 0, 0)$	-2
	$(0, 1, 0, 0, 0)$	-1
	$(0, 0, 1, 0, 0)$	0
	$(0, 0, 0, 1, 0)$	+1
	$(0, 0, 0, 0, 1)$	+2

Hello, and welcome to Lectures on Symmetry Optics. I'm Paul Mirsky. This is lecture 9, and the topic is: States of the Factor, Beam, and Grating

The singular factor is a basic element of symmetry optics. It's often drawn as a diagram with many open boxes and one filled box. More formally, it's represented by a vector which has many 0 entries and a single 1 entry. The factor also has some size – in this case, 5. This could represent a dark space 5 patches wide, containing a single bright patch. Every patch is always 1 wavelength wide.

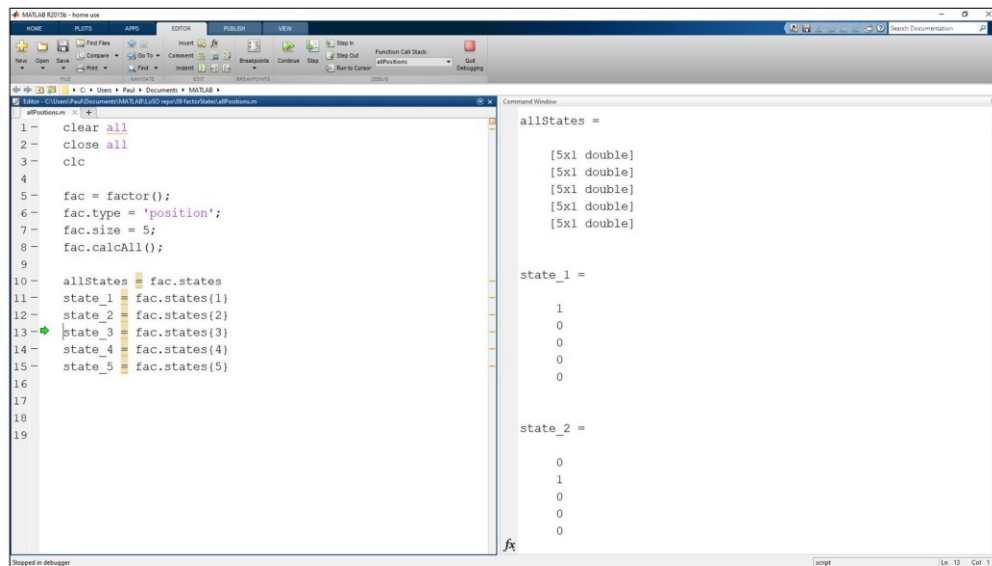
Up to now, we have always placed the 1 in the center of the vector by convention. However, we're now introducing the possibility of a putting the 1 at any location in the vector, not just the

center. The physical meaning is that the bright patch may be at any of the 5 possible positions within the space. The center of the space is position 0, and the patches to the right are +1 and +2. Those to the left are -1 and -2. For this reason, a singular factor can also be called a *position factor*.

The different vectors are called the *state vectors*, and the various positions are the states of the factor. The term 'state' is somewhat confusing. The term is borrowed from quantum mechanics, because the mathematical framework we're using is identical to that for a quantum particle which can be in 5 discrete positions. In this framework, the state vectors are all orthogonal to another, and they form a basis which spans an 5-dimensional complex vector space.

If you conceive of it as a particle, you would imagine that it might stay still *or* move around within the space. In this case, it's natural to call these various positions the 'different states' of the particle. But in optics, the pattern of light is stationary in time and is not moving around from one position to another. The patterns do change as the light propagates, but all of that is modeled without ever moving from one of these state vectors to another. For this reason, the word 'state' is somewhat awkward, but we will use it anyway. It's best to interpret it as the 'possible states'. We are not going to discuss quantum mechanics in much detail here. But historically, symmetry optics grows out of quantum mechanics through this connection. This also creates points of contact between Symmetry Optics, and more-established theory.

2 Companion code



```
1- clear all
2- close all
3- clc
4
5- fac = factor();
6- fac.type = 'position';
7- fac.size = 5;
8- fac.calcAll();
9
10- allStates = fac.states
11- state_1 = fac.states(1)
12- state_2 = fac.states(2)
13- state_3 = fac.states(3)
14- state_4 = fac.states(4)
15- state_5 = fac.states(5)
16
17
18
19
```

```
allStates =
[5x1 double]
[5x1 double]
[5x1 double]
[5x1 double]
[5x1 double]

state_1 =
     1
     0
     0
     0
     0

state_2 =
     0
     1
     0
     0
     0
```

In the companion code, the file *allPositions.m* shows a way to compute all of the state vectors for a position factor. We create a factor, and fill in two fields: the type is position, and the size is 5. Then we calculate all.

The results are contained in the properties of the factor object. We're going to assign these properties to variables just so that it prints onto the command window on the right. *AllStates* is an array of 5 vectors, each one is a column vector. We'll go through them one by one, and we see that the '1' is at a different location for each.

3 Plenary factors

Diagram

State vector



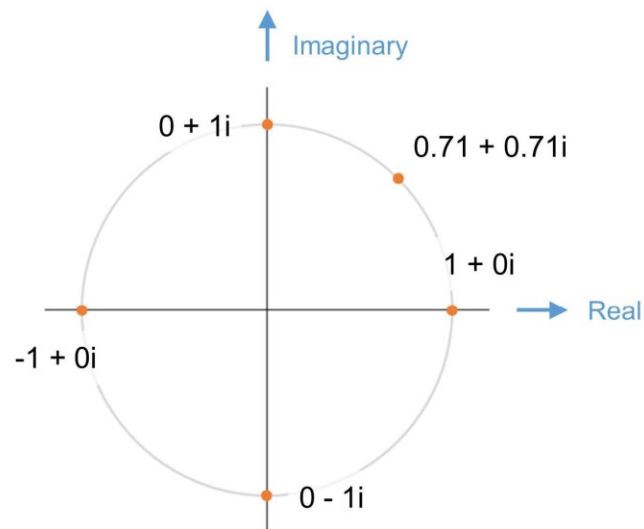
(1, 1, 1, 1, 1)

We're now finished introducing singular factors, or position factors as we're now calling them. Now, we'll do the same for plenary factors.

Previously, we considered only plenary factors which consisted of a vector with 1s for every entry. Implicitly, these have always been oriented perpendicular to the optical axis – in other words, at angle zero.

But before we can generalize to other angles, we first need to take a short detour.

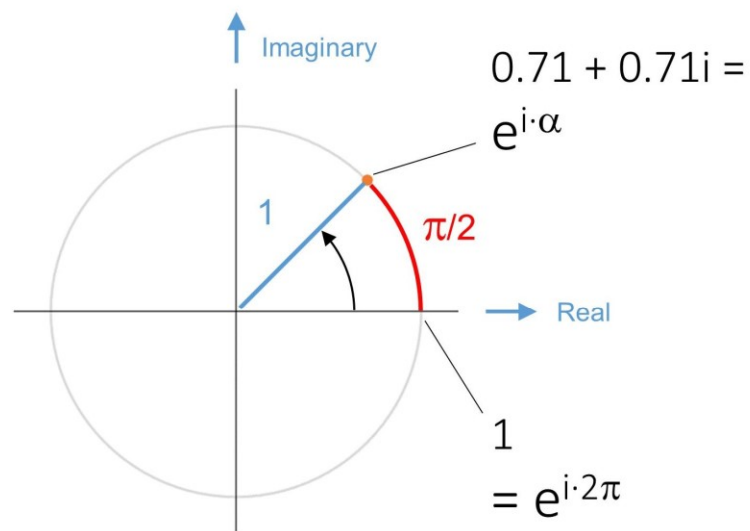
4 Unit circle



Up to now, we have only used two real numbers: 0 and 1. Now we're going to open up a much wider set of possibilities by using complex numbers.

In this diagram, the horizontal axis is the real component, and the vertical axis is the imaginary component. This point is $1 + 0i$, which is just 1. This point is $0 + 1i$, which is just i . This point is the sum of a real and an imaginary component. This is the unit circle. Every point on the unit circle has an absolute value of 1.

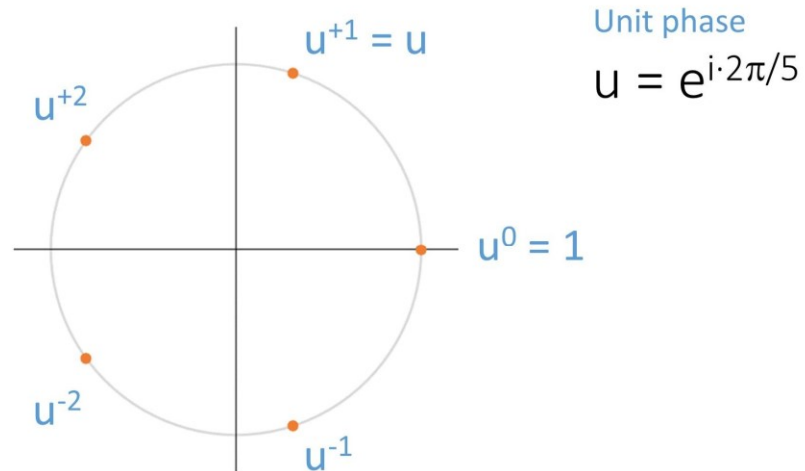
5 Phase angle



Every point also has a phase angle. The real number 1 is taken to define phase angle 0, and other angles are defined relative to it. Phase angle can vary between 0 and 2π radians. The way to visualize this is that the radius is 1, and the phase is the length of this circle segment measured along its own arc. When the length is 2π it will wrap fully around and return to zero. The phase angle itself is actually a real number, even though it corresponds 1:1 to a complex number.


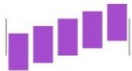



Note also that recently, in another lecture, we talked about angles on the unit circle in the context of provolution. But this type of phase angle is not describing provolution – it's a totally different application of the same mathematics. Anything that undergoes simple harmonic motion has a phase which varies in time.

6 Unit phase



Any factor has a *unit phase* u . In this example the factor size is 5. So the unit phase is 2π divided by 5 which is $1/5$ of a cycle around the circle. The only complex numbers we will consider are powers of this unit phase, which is the same thing as integer multiples of the corresponding angle. Unit phase with exponent 0 is 1, unit phase with exponent 1 is unit phase, then u^2 . But u^3 is actually the same thing as u^{-2} , then u^{-1} , then back to u^0 .

7 Plenary factors / Angle factors

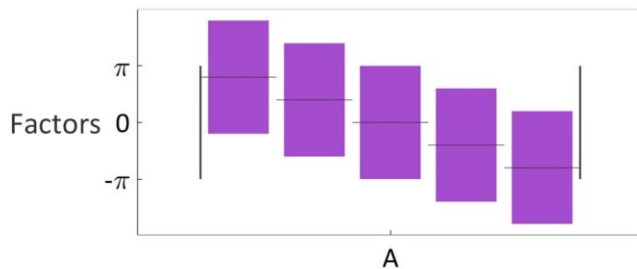
Diagram	State vector	Angle value
	$(u^{-1}, u^{+2}, u^0, u^{-2}, u^{+1})$	$-2/5$
	$(u^{+2}, u^{+1}, u^0, u^{-1}, u^{-2})$	$-1/5$
	$(u^0, u^0, u^0, u^0, u^0)$	$0/5$
	$(u^{-2}, u^{-1}, u^0, u^{+1}, u^{+2})$	$+1/5$
	$(u^{+1}, u^{-2}, u^0, u^{+2}, u^{-1})$	$+2/5$

Now that we have discussed complex numbers, we can introduce the more general case. Just as there exists a full basis of position vectors, there also exists a full basis of state vectors corresponding to all possible angles. For that reason, we will also begin referring to plenary factors as *angle factors* instead.

Tangentially, in the analogy with a quantum particle, these represent the various *momentum states* rather than angles.

All of the components of all the state vectors are various powers of the unit phase, so they all have absolute value 1. But the phases differ. For the zero-angle wavefront, all components are 1 which is the unit phase with a zero exponent.

8 Phase and factor diagrams



$$(u^{+2}, u^{+1}, u^0, u^{-1}, u^{-2})$$


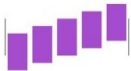



$$u = e^{2\pi i/5}$$

	Plane	Phases
This Diagram	Same	Different
Wavefront	Different	Same

To represent angle factors visually, we introduce a new aspect of factor diagrams in which the vertical dimension represents the phase angle. The bars on the left and the right indicate the range from $-\pi$ to $+\pi$ for a full range of one cycle. The phase angle of each patch is represented by the vertical location of the box, and you read from the midline of the box. So the center patch is at phase 0, while the patch at position +1 is at angle $-2\pi/5$. The diagram and the vector written below are two ways of representing the exact same information.

This diagram gives you an impression of a wavefront that is flat but is tilted to the right, and which will project rightward as it propagates. Casually, it's OK to think of it that way but you should know that it's actually a misinterpretation. If this were a wavefront, that would mean that all the points are at the same phase, and the shifts along the vertical axis would indicate that the left side of the wave is in a different plane, slightly ahead along the propagation axis. But actually all the points on this diagram are located precisely in the same plane, the front flat. And the vertical staggering indicates different phases. But it turns out that the wavefront and the phase-at-the-flat have the same values apart from a scale factor. So, for the sake of a convenient visualization, we can just think of it as a wavefront. Technically that's wrong, but it works out correctly anyway.

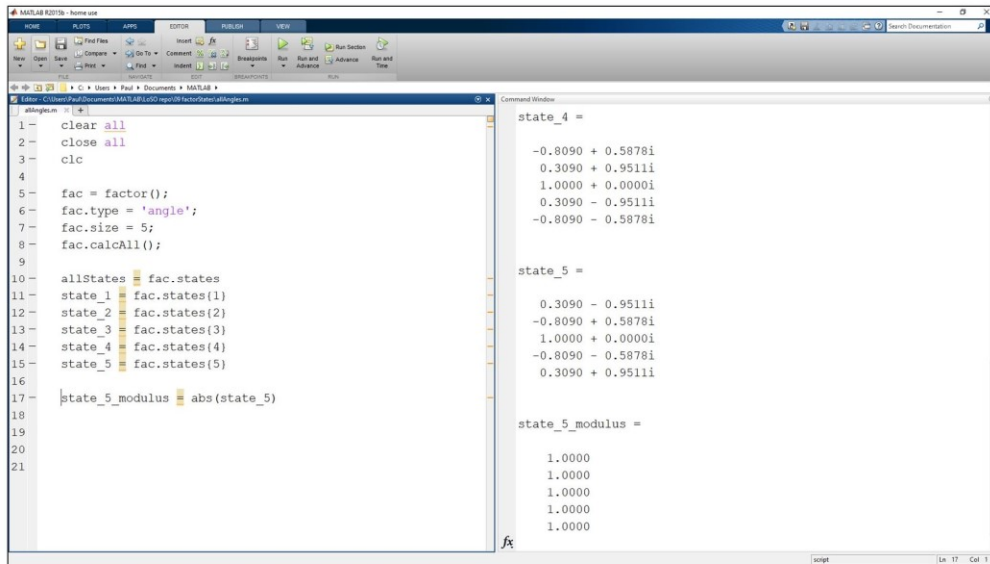
9 Plenary Factors / Angle factors

Diagram	State vector	Cycle value	Angle value
	$(u^{-1}, u^{+2}, u^0, u^{-2}, u^{+1})$	-2	$-2/5$
	$(u^{+2}, u^{+1}, u^0, u^{-1}, u^{-2})$	-1	$-1/5$
	$(u^0, u^0, u^0, u^0, u^0)$	0	$0/5$
	$(u^{-2}, u^{-1}, u^0, u^{+1}, u^{+2})$	+1	$+1/5$
	$(u^{+1}, u^{-2}, u^0, u^{+2}, u^{-1})$	+2	$+2/5$

Looking again at the basis vectors for angle factors, we can discern the trend. Angles range from $-2/5$ to $+2/5$ radians. The total angular range under consideration is 1 radian, meaning that the wavefront is propagating generally forward. If we scale the angles by factor of 5, which is the factor size, we get the same set of values that the corresponding position factors has – negative 2 to positive 2. The math actually works out to be a little cleaner if we scale the angle this way.

Even though we still call it the ‘angle’ value, the more-precise interpretation is the *number of complete cycles* around the unit circle that make up the state vector. This is easiest to see on the diagram. So at +1, the phase starts close to the maximum of $+\pi$ and decreases, eventually falling close to $-\pi$ which is the same as $+\pi$ where it began. That’s 1 complete cycle. At +2, the phase goes all the way around twice, etc., although it’s a little bit difficult to see the trend when the factor is only size 5. In all cases, the phase varies linearly as a function of position.

10 Companion code



```
1- clear all
2- close all
3- clc
4
5- fac = factor();
6- fac.type = 'angle';
7- fac.size = 5;
8- fac.calcAll();
9
10- allStates = fac.states
11- state_1 = fac.states(1)
12- state_2 = fac.states(2)
13- state_3 = fac.states(3)
14- state_4 = fac.states(4)
15- state_5 = fac.states(5)
16
17- state_5_modulus = abs(state_5)
18
19
20
21
```

```
state_4 =
-0.8090 + 0.5878i
0.3090 + 0.9511i
1.0000 + 0.0000i
0.3090 - 0.9511i
-0.8090 - 0.5878i

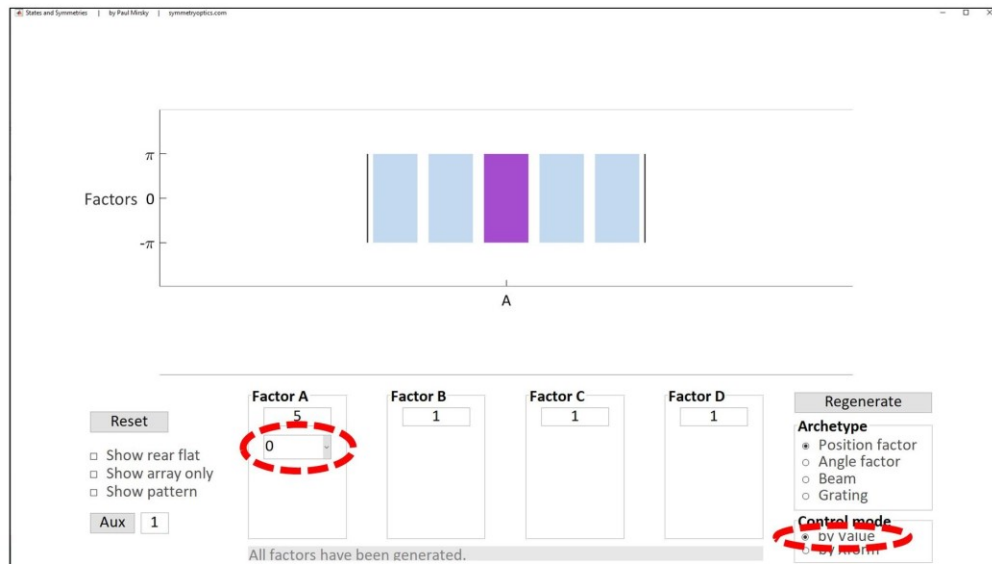
state_5 =
0.3090 - 0.9511i
-0.8090 + 0.5878i
1.0000 + 0.0000i
-0.8090 - 0.5878i
0.3090 + 0.9511i

state_5_modulus =
1.0000
1.0000
1.0000
1.0000
1.0000
```

Check out the companion code file *allAngles.m*. It's very similar to the position factor. We set up the factor object with a size and a type; this time the type is angle or plenary. It computes the 5 state vectors, each one is still 5 x 1.

The states look messy when they're written as complex numbers, except for state 3 which is all 1s. But even for a vector like state_5 which looks like this, if we take the absolute value we see that they are all modulus 1. The difficulty of interpreting these messy numbers is one big motivation for using the visual diagrams instead.

11 State and Symmetries app (SnS)



The companion code also includes a simple app called States and Symmetries, or SnS for short. You can download it from Github at <https://github.com/paulmirsky/symmetryOptics>; or, find the link at www.symmetryoptics.com/code. It's a very useful tool, and we'll use it throughout the rest of this lecture and the next. Let's see how it works. To start the app, open the file `SnS_start.m` and press run.

Let's look at the important elements of the screen. On the right, there's a panel marked *archetype*. There are four choices – *position factor*, *angle factor*, *beam*, and *grating*. Right now, 'position factor' is selected. Under the archetype panel, there's another panel marked *control mode*. The choices are *by value*, and *by transformation*. Right now 'by value' is selected, and we'll keep it that way for this entire lecture.

The screen shows panels for 4 factors – labeled factors A, B, C, and D – but because we've selected the 'position factor' archetype, only factor A is active right now. This box indicates that factor A is size 5, and the drop-down menu shows the various position values. When we select a value from the menu, it draws the state vector up on the canvas. Here's +2; here's -1, etc. Press *reset* to get back to 0.

Next we'll try a different size. Type 8 into the size box, and press *regenerate*. Now the canvas shows a vector with more dimensions and more values.

Next, we'll try an angle factor. Set the size back to 5, and in the archetype panel, select 'angle factor'. Press regenerate, and the app draws an angle factor. Again, the drop-down menu lets you select a value, and the app draws the state vector on the canvas

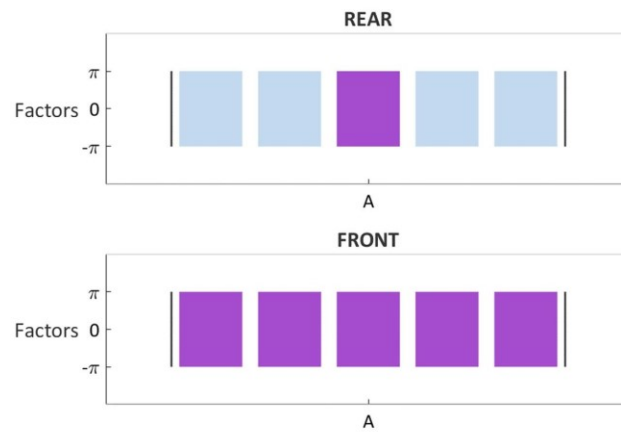
And now, a brief tangent. Traditionally, most lectures in optics feature a lot of analytic expressions and do most math symbolically. In contrast, the style of these lectures almost always uses numerical examples, and leans heavily on diagrams, and on code, and on apps, as you have been seeing. It would be easy to assume that these are inherent to the subject matter of symmetry optics. But in fact, that is wrong.

I believe it's possible and beneficial to use diagrams to make almost any technical subject easier without necessarily sacrificing rigor. Indeed, I believe that an intuitive understanding of a technical subject can be in some ways even superior to a purely formal one.

And obversely, symmetry optics could change as it grows and matures. I anticipate that it will become far more abstract. It could outstrip our ability to represent faithfully through diagrams.

Now, back to the lecture.

12 Front and rear flat, angle

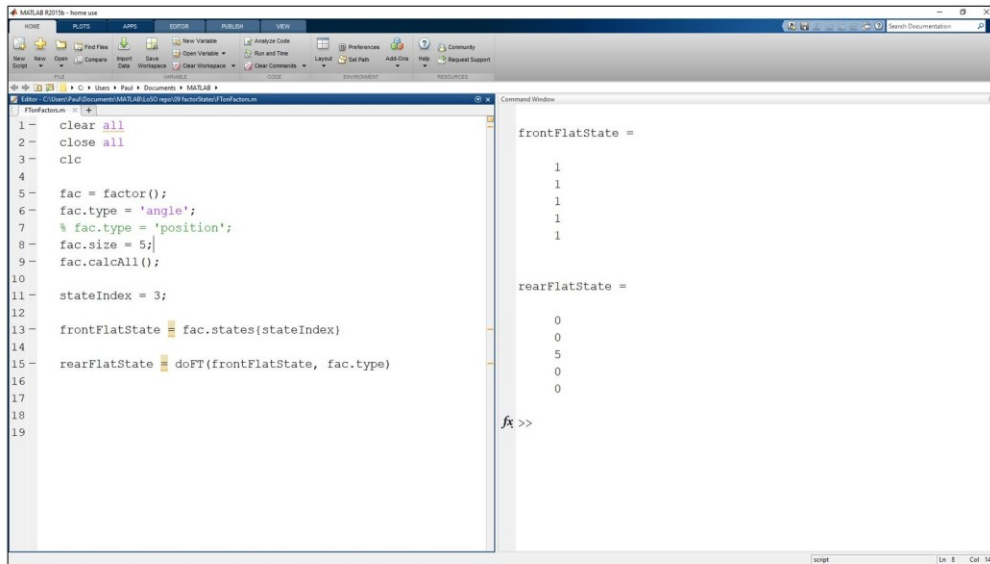


Next, we'll imagine that the factor we're setting lies in the front flat of the lens-limited configuration, and we'll add the rear flat. To do this, we'll check show rear flat, and then click reset.

We see that the front flat is an angle factor, and the rear flat is a position factor. We already know that the rule for computing the Fourier transform is to reverse the order of the factors, and to invert the type. In this case there is only one factor, so reversing the order of factors doesn't do anything. But the type is inverted from front to rear.

That formulation of the Fourier transform was OK when we were only considering a single state for each factor type. But it won't work now that we have many different state vectors for each type.

13 Fourier transform in comp code



The screenshot shows the MATLAB R2015b interface. The script editor on the left contains the following code:

```
1- clear all
2- close all
3- clc
4
5- fac = factor();
6- fac.type = 'angle';
7- % fac.type = 'position';
8- fac.size = 5;
9- fac.calcAll();
10
11- stateIndex = 3;
12
13- frontFlatState = fac.states[stateIndex]
14
15- rearFlatState = doFT(frontFlatState, fac.type)
16
17
18
19
```

The Command Window on the right shows the output:

```
frontFlatState =
    1
    1
    1
    1
    1

rearFlatState =
    0
    0
    5
    0
    0

fx >>
```

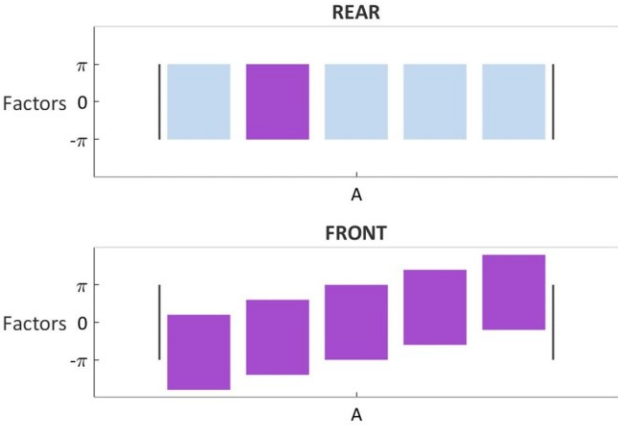
Instead, we just use the ordinary Discrete Fourier Transform in place of the original rule to invert the type. Actually, the original rule was really only ever just a very narrow special case of the Fourier transform.

Let's see an example in the companion code. Open the file *FTonFactors.m*. First we create an angle factor. Then we choose one of the state vectors to be at the front flat. We apply the Fourier transform to that vector, and it gives us the state vector for the rear flat.

We can see from this that actually the state vector is scaled from what we ordinarily expect from position factors, because instead of many 0s and a 1, we have many 0s and a 5. Actually, symmetry optics is like quantum mechanics in that the overall length of the state vector doesn't really matter. If we allow for a scale change, a single 5 is the same as a single 1.

Let's run the code again, but this time we will step inside the function *doFT()*. This is a wrapper for Matlab's function *fft*. A little bit of shifting needs to be done to correct for the fact that our vectors have 0 at the center, while Matlab's 'fft' has 0 at the left.

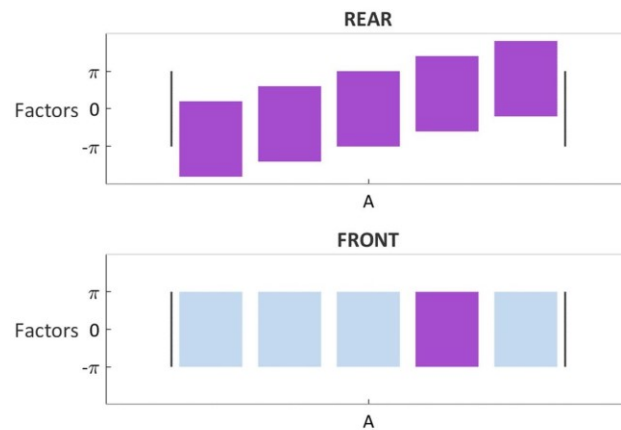
14 Front and rear flat, position



When we plot these state vectors in the app, we see that it works very intuitively. The angle at the front flat determines the position at the rear flat. When the angle at the front flat is 0, the position at the rear flat is zero. When the angle at the front flat is +1, the position at the rear flat is +1.

In other words, the value of the factor remains constant from the front flat to the rear flat. You also get the visual impression of a tilted wavefront that is steering the light towards a certain position.

15 Front and rear flat, angle



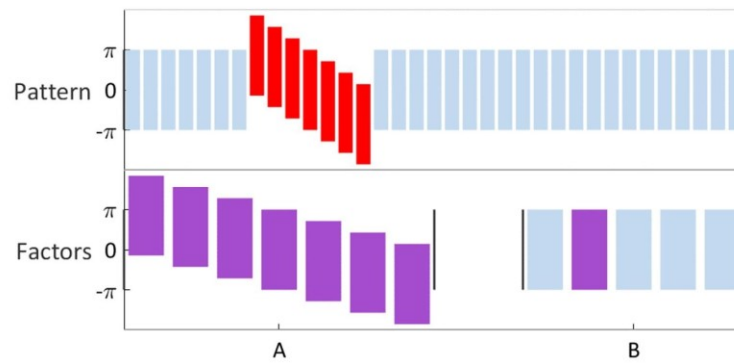
The whole thing also works in reverse. To see this, change the archetype back to position factor, and regenerate.

A single patch of light expands with a wide beam of a full radian, and it takes up the full wide limit at the rear flat. But, if we shift the starting patch to other positions at the front flat, it ends up at different angles at the rear flat. Recall that there is a lens in between the two planes, and so the tilt at the rear flat is coming because the starting patch moves off of the axis of the lens.

The factor value is still conserved from one plane to the other, but in this case there is a small quirk. The position at the front flat is +1, but the angle at the rear flat appears to be -1. We need to interpret the angles in the rear plane with the sign or sense reversed. Conceptually, the factor value is remaining the same.

By the way, note that everything to do with complex state vectors only works at the front and rear flats. That includes everything in this lecture and the next, and everything computed with the States-and-symmetries app. For any plane in between the flats, you can't use the complex vectors. One reason is that they have no means of representing roundness. Perhaps the theory could be extended to make source-target grids from complex vectors, but that's currently unknown.

16 Beam in factors



The next step in our development is to consider the next-simplest system: the beam. As part of its definition, it has 2 factors, A and B. In the Archetype panel, click 'beam', and set the size of Factor A to 7 and the size of Factor B to 5, then un-click 'show-rear-flat', and regenerate.

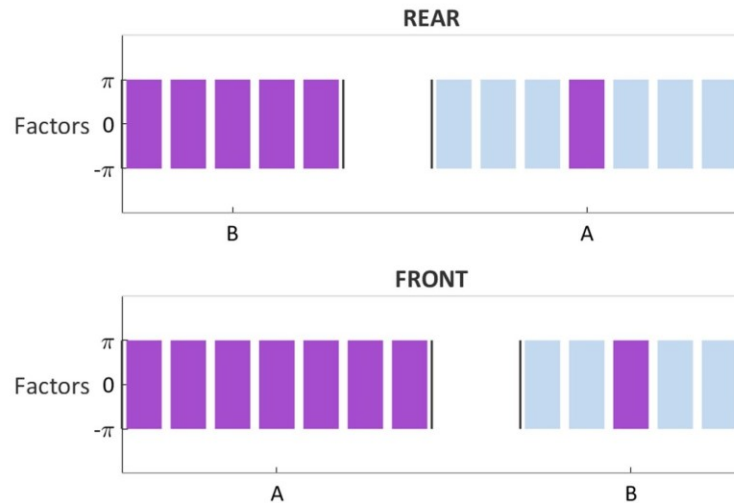
Here is the factor diagram for a beam. We've seen this type of diagram before, many times. Now we'll try a new capability of the app, which is to draw the spatial pattern in addition to the factors. Earlier we were studying only one factor, and in that special case the factor and spatial diagrams look identical, so we didn't bother to draw them separately. But with two or more factors, the pattern looks different and must be drawn separately. Check show pattern and click reset. Now we see the beam drawn more-or-less as it would appear in space, with a bright stripe and dark space on either side, and some finite wide limit.

We've discussed the beam many times in earlier lectures, but all of those cases were limited to 0 angle and 0 position. But now, with many possible states of the factors, there also exist many possible states of the beam. Position factor B determines the position of the beam within the wide limit. These are the five different possible positions, and we can see that the beam and the factor correspond. Note that we don't consider any other positions; for example, it's not possible for the beam to be just one or two patches off-center; the states are separated from one another by a full stripe width. Otherwise, they overlap too much and don't count as a distinct state.

The beam can also be at any angle, depending on the state of the angle factor. The beam can be at 7 possible angles.

Also, the positions and angles can be varied independently, in any combination. There are A possible angles and B possible positions, for a total of $A \cdot B$, or this in case $7 \cdot 5$ or 35 possible states of the beam.

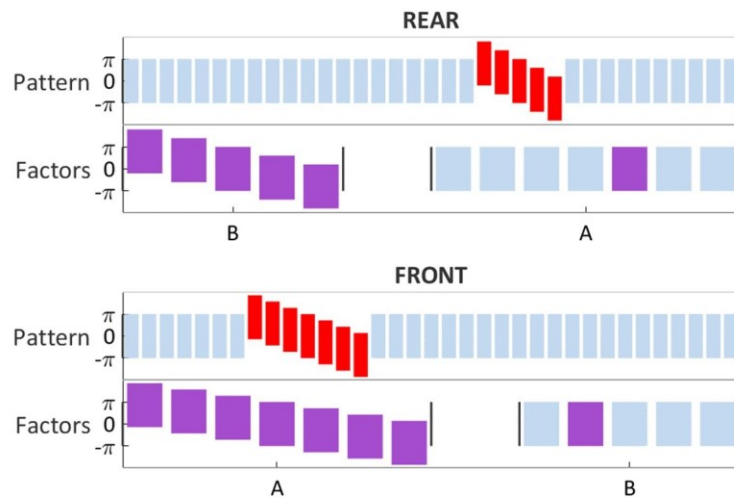
17 The beam, front and rear flats



Next let's turn off the spatial diagram but look at front and rear flats. This also is familiar from earlier lectures, and it shows the reverse-and-invert rule more clearly than a single factor can. The front flat is A then B, the rear flat is B then A. In the front flat, A is angle type and B is position type. In the rear flat, A is position type and B is angle type.

Now we also consider all the possible states for both factors. The rule is simply that we apply the Fourier transform to each factor individually. What we find is that both factor values remain the same between the two planes, just as the factors would individually. If the beam angle at the front flat is +1, then the beam position at the rear flat is also +1. If the beam position at the front flat is -1, then the beam angle at the rear flat is also -1, although once again we must reverse the sign when we interpret the answer that the app computes.

18 Beam states, front and rear flats

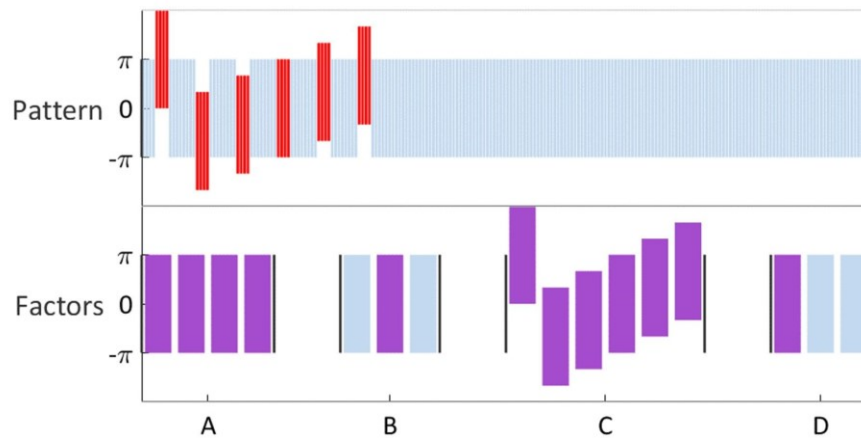


Now, we'll bring back the pattern and see all the information about the beam at once. Looking mainly at the spatial pattern, the beam behavior is pretty intuitive: The angle of the beam in the front determines the direction of propagation, which determines the position in the rear focal plane.

It's essentially the same thing if you do it in reverse: the position of the beam in the front plane determines the angle in the rear plane. You can think of this in two ways: firstly, everything has to work if you flip it front-to-rear, so you can think of this as the angle in reverse. The second is to think of rays of light. If they are parallel to the lens axis but offset from the lens axis, they will each get angled by the lens by a different amount, and they will all get focused to a point, but will arrive there from different angles.

You can vary position and angle independently, and any combination is possible.

19 Grating states



The final topic of the lecture is the grating archetype. For the factor sizes, we set 4, 3, 6, and 3. We also de-select the rear flat, and regenerate. Right now all factors are at the zero-value state, so the factor chain and pattern look the same as they did in earlier lectures.

But just as with the beam, we can choose other states. We'll start with factor A. If we think of the grating as an array of identical small beams, then factor A works just like it does in the ordinary beam, and determines the angle. The same angle is repeated for all of the small beams.

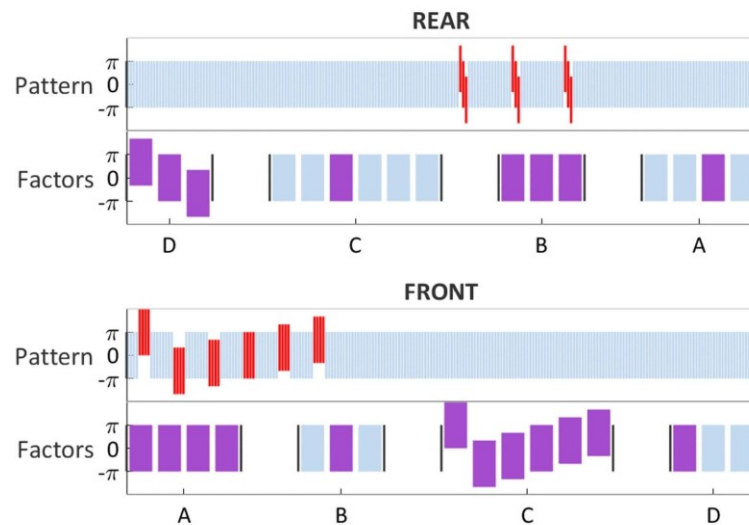
Factor B determines the position of the beam within the period. The period in this case is 3 times as wide as the small beam, so it can be in any of the 3 positions within the period. It's easy to misinterpret this, because it looks like the entire pattern is being shifted. That's not quite correct. To see it more clearly, click on *Show array only*. The array consists of 6 periods, each one 12 patches wide. For example, the first period goes from the very left of the array to this point halfway between the first two stripes. Position factor B determines the position of the stripe within each period. The array as a whole does *not* move. The periods, also, do *not* move. The first period still starts at the left side of the array and extends for 12 patches. But now, the stripes are positioned in the right third of the period, instead of being in the middle of the period.

Angle factor C applies a different phase factor to each stripe. But notice that within any given stripe, all the patches are at a single phase. This is different from angle factor A, which applies a different phase factor to each patch within the stripe.

Finally, position factor D sets the position of the array within the wide limit. Note that there are only 3 possible positions for the array. For example, the array can be at position 0 or position +1, but it can't be halfway in between. This may seem unphysical, but the right interpretation is that two different arrays that mostly overlap are actually counted as the same pattern. It's only when it moves by one full array width that we count it as a distinct state.

These four degrees of freedom are independent from one another and combined in arbitrary ways. Together, they allow us to specify any possible state of the grating.

20 Grating states, front and rear flats



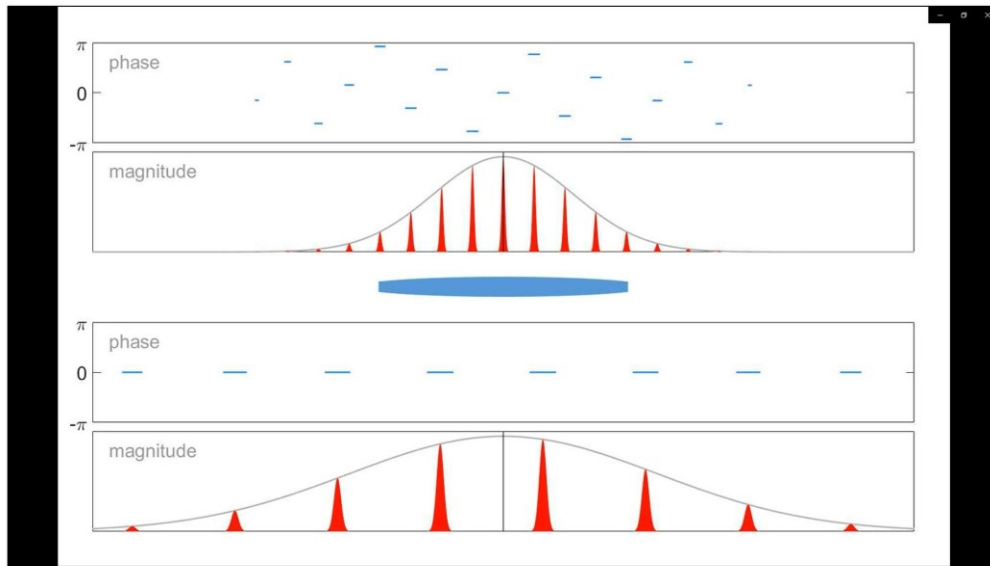
Finally, let's examine the grating in two different planes. The angle of factor A at the front flat determines the position of factor A at the rear flat. In other words, it's the angle of the individual small beams at the front that determines the overall position of the array at the rear. This makes physical sense

Contrast that with the angle of factor C. Just like factor B in the previous slide, it does not move the position of the array itself, but rather moves the position of the small beams within each period.

Position factors B and D have a similar effect but in reverse. Factor D sets the position of the array at the front flat, and this determines the angle of the small beams at the rear flat, after passing through the lens. Factor B is less obvious. Shifting the position of the front-flat stripes causes phase shifts at the rear flat, which are applied uniformly to all the patches within a single stripe. This is somewhat counterintuitive, because the precise location of grating slits is rarely taken into account in practical optics. Usually, it's not considered important. But this shows that it does have a physical effect.

As we mentioned earlier, we take the FT of each factor individually, before we form their outer product to get the pattern. So in order to compute the FT, we incorporate our prior knowledge about the factors. In a sense, this is cheating a little bit. The lens-limited configuration can't somehow compute the FT differently for different patterns. It simply treats any arbitrary input pattern as a single long state vector.

21 Gaussian grating, computational



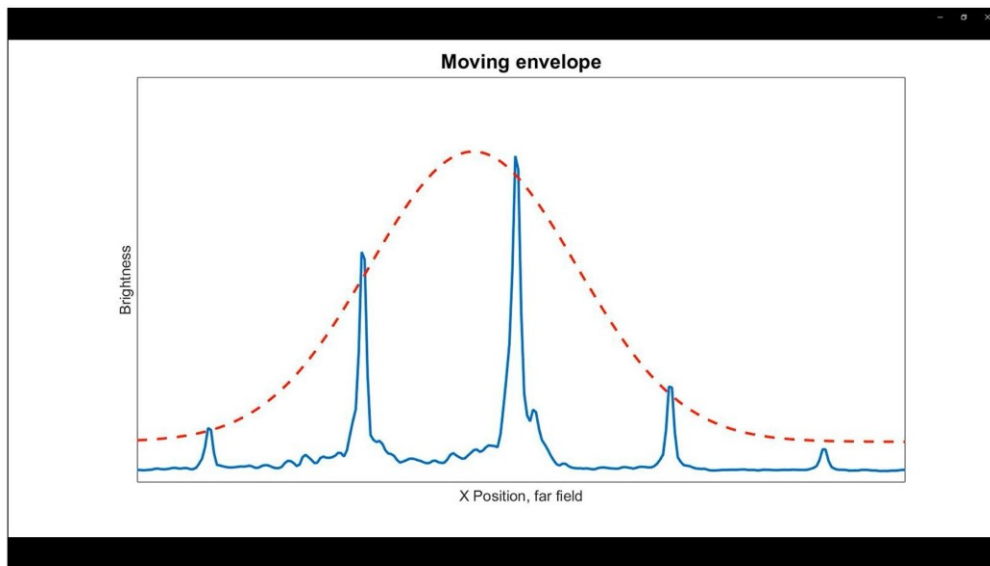
But actually, it still gets the answer right. To see that it works, check out *oneGaussianGrating.m* in the companion code. This file doesn't actually use symmetry optics at all. Instead, we just create a complex vector where the magnitudes are formed from Gaussian distributions.

The magnitude is in red, and the phase is on a separate axis in blue. We specify the front state, which is shown on the lower two axes, and it computes the rear state and plots it on the top two axes. This is just the standard Matlab FFT computation. We're not telling it anything about the factor structure of the pattern. It just treats the whole thing as one long vector.

If we sweep the array to various positions at the front, it changes the angle of factor D at the rear.

Likewise, if we sweep the stripe to various positions at the front, it changes the angle of factor B at the rear.

22 Gaussian grating, experimental



The same thing works in reverse. In other words, if we sweep the angles in the front, it changes the positions at the rear. This angle-only change can be implemented experimentally using a *spatial light modulator*. In this experiment, we are modulating the angle of factor A in the front, and we see the array changing position in the rear.

In this experiment, we are modulating the angle of factor C in the front, and we see the stripe changing position in the rear.

Links to these videos can be found on symmetryoptics.com, near the link to this video.

23 Reviewing key points

That's all for this lecture, so let's review the key points:

- A size- n position factor (aka a singular factor) can be in any of n different position states, represented by the location of the single '1' in the state vector.
- A size- n angle factor (aka a plenary factor) can be in any one of n different angle states, represented by the number of phase cycles in the state vector.
- From the front flat to the rear flat, each individual factor undergoes the Fourier transform independently. The position value is conserved and becomes the angle value (scaled), or vice-versa.
- The beam and the grating can both exist in many different states, determined by a combination of position and angle factors.

24 Outro

I hope you've found this class informative and interesting. To learn more about symmetry optics, please check out www.symmetryoptics.com. If you have specific questions about this or other lectures, please post them on Reddit at www.reddit.com/r/symmetryOptics/ and I'll try to answer them.

This is a new field, and there's a lot of opportunity to discover new science and develop new applications. I hope you'll take advantage.

I'm Paul Mirsky, thanks for listening.